Performability Modelling of Homogeneous and Heterogeneous Multi-server Systems with Breakdowns and Repairs

Enver Ever

School of Computing Science

Middlesex University

A thesis submitted to Middlesex University in fulfillment of the requirements for degree of Doctor of Philosophy

November 2007

Dedicated to, Süleyman and Hatice EVER

Abstract

This thesis presents analytical modelling of homogeneous multi-server systems with reconfiguration and rebooting delays, heterogeneous multi-server systems with one main and several identical servers, and farm paradigm multi-server systems. This thesis also includes a number of other research works such as, fast performability evaluation models of open networks of nodes with repairs and finite queuing capacities, multi-server systems with deferred repairs, and two stage tandem networks with failures, repairs and multiple servers at the second stage. Applications of these for the popular Beowulf cluster systems and memory servers are also accomplished.

Existing techniques used in performance evaluation of multi-server systems are investigated and analysed in detail. Pure performance modelling techniques, pure availability models, and performability models are also considered. First, the existing approaches for pure performance modelling are critically analysed with the discussions on merits and demerits. Then relevant terminology is defined and explained. Since the pure performance models tend to be too optimistic and pure availability models are too conservative, performability models are used for the evaluation of multi-server systems. Fault-tolerant multi-server systems can continue service in case of certain failures. If failure does not occur at a critical point (such as breakdown of the head processor of a farm paradigm system) the system continues serving in a degraded mode of operation. In such systems, reconfiguration and/or rebooting delays are expected while a processor is being mapped out from the system. These delay stages are also taken into account in addition to failures and repairs, in the exact performability models that are developed. Two dimensional Markov state space representations of the systems are used for performability modelling. Following the critical analysis of the existing solution techniques, the Spectral Expansion method is chosen for the solution of the models developed.

In this work, open queuing networks are also considered. To evaluate their performability, existing modelling approaches are expanded and validated by simulations, for performability analysis of multistage open networks with finite queuing capacities. The performances of two extended modelling approaches are compared in terms of accuracy for open networks with various queuing capacities.

Deferred repair strategies are becoming popular because of the cost reductions they can provide. Effects of using deferred repairs are analysed and performability models are provided for homogeneous multi-server systems and highly available farm paradigm multi-server systems.

Since one of the random variables is used to represent the number of jobs in one of the queues, analytical models for performance evaluation of two stage tandem networks suffer because of numerical cumbersomeness. Existing approaches for modelling these systems are actually pure performance models since breakdowns and repairs cannot be considered. One way of modelling these systems can be to divide one of the random variables to present both the operative and non-operative states of the server in one dimension. However, this will give rise to state explosion problem severely limiting the maximum queue capacity that can be handled. In order to overcome this problem a new approach is presented for modelling two stage tandem networks in three dimensions. An approximate solution is presented to solve such a system.

This approach manifests itself as a novel contribution for alleviating the state space explosion problem for large and/or complex systems. When two state tandem networks with feedback are modelled using this approach, the operative states can be handled independently and this makes it possible to consider multiple operative states at the second stage.

The analytical models presented can be used with various parameters and they are extendible to consider systems with similar architectures. The developed three dimensional approach is capable to handle two stage tandem networks with various characteristics for performability measures. All the approaches presented give accurate results.

Numerical solutions are presented for all models developed. In case the solution presented is not exact, simulations are performed to validate the accuracy of the results obtained.

Acknowledgements

I gratefully thank my director of studies Dr. Orhan Gemikonakli for introducing me to this area of research and for his outstanding supervision, throughout this research. Dr. Gemikonakli has contributed in every stage of my PhD. studies. The technical discussions and guidance have been highly motivating, productive, and have given me a great deal of professional satisfaction. This work would not be possible without his precious contributions.

I also thank my second supervisor Prof. Ram Chakka for his outstanding supervision. The theoretical support he provided was very useful and helped me understand many concepts in performance, availability, and performability modelling. I am thankful to his support and guidance.

My third supervisor Dr. Ferdinand Katsriku was very supportive and guided me throughout this research.

Dr. Altan Koçyigit was very helpful and he has provided very useful discussions and support especially for the simulations.

My office friends were always with me when my computer crashed and my laptop broke down. I thank Mrs. Rajan, Dr. Foster and Mr. Thakker for their support.

I also thank my sister and her husband for their support.

During difficult times Ms. Yoney Kırsal was always with me with her support and altruism.

List of Publications

The work presented in this thesis has given rise to the following publications.

1. Ever, E., O. Gemikonakli, R. Chakka and T.V.Do (2005), "The Use of Markov Modulated Poisson Process (MMPP) in Open Queuing Systems with Breakdowns and Repairs", In Proceedings of PREP 2005, Lancaster.

2. Gemikonakli, O., T.V. Do, R. Chakka and E. Ever (2005), "Numerical Solution to the Performability of a Multiprocessor System with Reconfiguration and Rebooting Delays", In Proceedings of ECMS 2005, Riga, Latvia.

3. Ever, E., O. Gemikonakli, R. Chakka and T.V. Do (2005), "A Mathematical Model for Performability Evaluation of Heterogeneous Multiprocessor Systems with Reconfiguration and Rebooting Delays", In Proceedings of ESM 2005, Porto, Portugal.

4. Ever, E., O. Gemikonakli and R. Chakka (2006), "A Mathematical Model for Performability of Beowulf Clusters", In IEEE Proceedings of 39th Annual Simulation Symposium, Huntsville, AL, USA, April.

5. Ever, E., O. Gemikonakli and R. Chakka (2006), "A Stochastic Model for Highly Available Clusters with One Head and Several Identical Computing Nodes", In Proceedings of The 9th International Conference on Computer Modelling and Simulation, UKSIM 2006, Oxford, U.K., April.

6. Chakka, R., E. Ever and O. Gemikonakli (2006), "Modelling Open Networks with Breakdowns, Repairs and Finite Buffers, Using an IPP Departure Process Model", In Proceedings of the 10th WSEAS International Conference on Computers, Athens, Greece, July.

7. Chakka, R., E. Ever and O. Gemikonakli (2006), "An Approach to Modelling Open Networks with Unreliable Servers and Finite Buffers", WSEAS Transactions on Computers, Vol 5, Issue 11, November, ISSN 1109-2750.

8. Ever, E., O. Gemikonakli and R. Chakka (2006), "Performability Analysis of Highly Available Clusters with Break-downs and Deferred Repairs", In Proceedings of Fourth International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks, (HET-NETs'06), Ilkley, UK, September.

9. Ever, E. and O. Gemikonakli (2006), "Mathematical Modelling for Performability Analysis of Homogeneous Multi-server Systems with Deferred Repairs", In Proceedings of European Modelling Symposium 2006, London, U.K. September.

10. Gemikonakli, O., G. Mapp, E. Ever and D. Thakker (2007), "Modelling Network Memory Servers with Parallel Processors, Break-downs and Repairs", In IEEE Proceedings of 40th Annual Simulation Symposium, (ANSS'07), Norfolk, USA, March.

11. Chakka, R., E. Ever and O. Gemikonakli (2007), "Joint-State Modelling for Open Queuing Networks with Breakdowns, Repairs and Finite Buffers", In IEEE Proceedings of 15th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTs'07), Bogazici University, Istanbul, October.

12. Ever, E., O. Gemikonakli and R. Chakka (2007), "Analytical Modelling and Simulation of Small Scale, Typical and Highly Available Beowulf Clusters with Breakdowns and Repairs". Journal of Simulation Modelling Practice and Theory (International Journal of the Federation of European Simulation Societies - EUROSIM Former title: Simulation Practice and Theory: Under Review).

13. Gemikonakli, O., Ever E., and A. Kocyigit (2007), "Approximate Solution for Two Stage Open Networks with Markov-modulated Queues Minimizing the State Space Explosion Problem". Journal of Computational and Applied Mathematics (ACCEPTED FOR PUBLICATION).

14. Gemikonakli, O., E. Ever and A. Kocyigit (2007), "Performability Analysis of Multi-Server Systems with Deferred Repairs and Reconfiguration or Rebooting Delays", International Journal of Communication Systems (Under Review).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Developments in the computer field have increased rapidly since the late 1960s. Engineers and scientists were faced with the fact that users should share data and Input/Output devices through an environment. As the usage of computer systems and their connections through networks has increased rapidly, many large and interesting collections of materials became available electronically. This makes both users and designers aware of the requirement of sharing data and resources with ease, and affordable performance and reliability measures.

Today, computer and communication systems are very widely used in research, industry, business and also in the everyday life of people all over the world. These systems are used at cash points, travel reservation systems, air traffic controls, hospitals' diagnostic equipment, patient monitoring systems, transportation industry, parallel computing, and scientific research. As the users' demands increase, the complexity of computer networks, network products, communication systems and information systems also increases. As a result of this, it becomes more difficult to understand and handle various components of systems which interact in these complex environments, and to make sure that all the implications have been covered and considered properly.

When the systems are implemented without performance evaluation, there is a risk of consuming resources by committing them to the development of a system or product which does not meet optimum requirements. Also it is possible to end up with a system with inadequate performance and/or availability characteristics. In the modern world, with fast improving technology, highly competitive research and industry environment, such a risk is not acceptable.

Performance and availability analysis using modelling, allows engineers, researchers, developers and users to predict and detect possible drawbacks of the systems. This provides early correction and accurate planning. Modelling systems for performance and availability

analysis is helpful for understanding the complex interaction of systems' components and elusive effects of various factors (Jain 1991, Law and Kelton 2000). Performance analysis is also useful for optimisation of various system characteristics. Results obtained, enables system designers to examine optimal design tradeoffs. Performance and availability analysis of many systems becomes essential for the success or failure of many projects (Jain 1991, Banks et. al. 2005).

Multi-server computer and communication systems are commonly preferred because of the greater computation power they can provide, and improved reliability. Multi-server systems are more reliable than single server ones since server failures do not cause complete system failures, and the system can continue serving in a degraded mode. Multi-server systems are also important for parallel processing as well.

A fault-tolerant multi-server system needs to be evaluated for design and effective usage concerns. Both performance and availability evaluation is important for designing and employing the best system that meets the user requirements. The use of performance and availability evaluation techniques before such a system is physically implemented, would be very useful in terms of understanding the system's general behaviour and modifying the system to meet service level agreements in terms of system performance and availability.

1.1 Performance Evaluation Techniques

There are three techniques used for performance evaluation of computer and communication systems. These are benchmarking, simulation, and analytical modelling (Jain 1991, Banks et. al. 2005).

The process of performance analysis by actual measurements is called benchmarking or measurement and the input workloads used are the benchmarks. Benchmarking gives very accurate results. However, these results are of limited use since extrapolation of these results to suit changes in system or workload is usually not possible. Benchmarking is only possible if something similar to the proposed system already exists. If the concept considered is new, analytical modelling and simulation are the only techniques available. Benchmarking is also, usually very costly in terms of equipment, personnel, and time (Jain 1991).

Analytical modelling and simulation enables performing analysis more quickly and cost effectively, without the need for prototyping. This is very important for identifying and solving performance and availability problems that would appear otherwise later in the design process.

Simulation is mimicking the operation of a real world process over time (Banks et. al. 2005). Simulation process involves building a simulation model of the system considered. This model can be validated against existing systems and then altered to reflect the proposed modifications. This approach is very flexible, and it gives fairly accurate and acceptable results, but for sufficient accuracy, simulations require relatively high computation times (Law and Kelton 2000). Both benchmarking and simulation are experimental approaches and they are costly especially in terms of time.

Analytical modelling gives rise to formulae and/or numerical procedures that are computationally more efficient compared to simulations (Law and Kelton 2000, Banks 2005). Analytical modelling of a system, or a specific part of it requires less computation. Usually the goal of performance and availability studies are to compare different alternatives, or to find the best architecture in terms of performance and cost effectiveness, or to find the optimal parameter values. When simulations are employed it may be possible to search the space of parameters and their interactions for the optimum combination, but the tradeoffs among the parameters and among performance measures may not be clear in a few runs, and this would normally take long time. If benchmarking is used, it will not be easy to tell if the result is due to a particular parameter or a change in the environment. Analytical models generally provide the best information for the effects of various parameters and their interactions (Jain 1991). However, analytical modelling requires a relatively high level of mathematical skills. Also analytical modelling sometimes requires a degree of assumptions to simplify the systems considered (Jain 1991, Trivedi 2002, Banks 2005). Analytical modelling approach is very widely used in computer science for performance, availability, and reliability evaluation of complex computer and communication systems. This approach is ideal for quick and, once validated, for relatively accurate results (Trivedi 2002, Banks 2005).

The models provide either exact or accurate approximate solutions for the steady state probabilities of the systems considered. Once the steady state distributions are computed, it is possible to obtain various performability measures. There are some hierarchical approaches which can be employed for performability evaluation of such systems (Ciardo et. al. 1990, Trivedi et. al. 1990, Temsamani and Carrasco 2002). However these models mainly use reward rates for a specific performability measure of interest. Because of this, some difficulties are expected in providing reward rates especially for specific states where the system is not providing service. Also, the relationship between hierarchical levels cannot be established successfully. It is desirable to provide solutions to find exact or approximate steady state probabilities.

For the two stage tandem networks, since both of the random variables are used to represent

the number of jobs in each stage, these systems cannot be considered for performability measures. Because of the same reason it is also not possible to consider multiple servers at the second stage. It is desirable to provide an approximate and accurate solution which can make it possible to consider multiple operative states at the second stage.

1.2 Scope of Investigation

This research project aims to develop analytical models for evaluation of performance and availability measures of various multi-server systems. Considerable focus is on obtaining the most effective and accurate solution for systems which are prone to failures and repairs. In this study analytical modelling techniques are used to model various complex multi-server systems. Analytical methods are employed together with certain assumptions in order to achieve a certain degree of mathematical tractability. Techniques such as probabilistic analysis (to represent the behaviour of the system under study), Markov processes and queuing theoretic models are used extensively.

The considered multi-server systems are prone to failures and repairs. For such systems availability studies are required in order to specify the ability of the system considered to be in a state to perform an operation at a given instant of time. On the other hand, another important issue for such systems is the performance of the available servers which is essential for handling the incoming jobs. A more realistic analysis method which is called performability analysis is used in this study. Performability analysis is introduced in (Beudry 1978) and a conceptual framework of performability has been considered by Meyer in (Meyer 1980). Performability models are used to combine performance and availability/reliability concerns of multi-server systems and such an approach is more realistic since multi-server systems are often considered for fault-tolerant applications.

Homogeneous multi-server systems have been considered together with failures and repairs in literature. Exact solutions are provided for various performability measures. These studies assume that a failed processor can be mapped out of the system and a repaired processor can be readmitted to a system without causing any delays. However, in case of server failures in practice some delay is expected when a failed processor is being mapped out from the system or a repaired processor is being admitted into the system. In (Trivedi et. al. 1990) reconfiguration and rebooting delays are introduced in order to take such delays into account. However, in that study, queuing characteristics of a multi-server system is not considered. Homogeneous multi-server systems with reconfiguration and rebooting delays should be considered together with queuing issues such as finite, infinite queuing capacities,

job losses due to the blocking behavior of bounded queues, the mean number of jobs in the queue and system etc., for accurate and useful predictions. This is one of the research items of this thesis.

Farm paradigm architecture is another well known configuration used for multi-server systems. This architecture is commonly used since it provides ease of use for multi-server systems with parallel computation facilities (Wagner et. al. 1997). Pure performance models and availability studies of such systems are given in the literature. However, performability measures should be used for evaluation of such systems, since they are also fault-tolerant. Furthermore, the single point of control and various service facilities that can be provided makes the performability analysis of these systems even more interesting and essential. It is possible to consider reconfiguration and rebooting delays in order to make the performability models more realistic.

Another interesting and commonly used multi-server architecture is open queuing networks. These models can be used for computer networks, and modern communication systems with inter-connected nodes. For open queueing systems with relatively small number of nodes, interrupted poisson process (IPP) and joint state approximate models are presented in (Chakka 1995, Chakka et. al. 2000). However, in these studies, it was assumed that the nodes have unbounded queueing capacities. This unrealistic assumption puts great limitations to the practical use of these models as they are. It is possible to extend these studies and present performability models for open queuing systems where the queues at nodes have indeed finite capacities. Also it is desirable to compare the performances of these two approaches in terms of accuracy for systems where the queues at each node can have various capacities. This is succesfully addressed in this thesis as evidenced by simulation studies.

Deferring the server repairs significantly affects the performability to cost ratio in multi-server systems. As the system's complexity, the number of components in the system, and cost of the repairs increase, traditional repair strategies may become expensive (Sun et. al. 2005, Carrasco 2006). Deferred repair strategies are commonly used in order to reduce these costs. Analytical models are presented for performance and reliability studies of multi-server systems using deferred repair strategies, but queuing issues are not considered in detail. Usually a threshold value is defined in order to specify the minimum number of servers which should remain in the system before the repairman is called (Tang and Trivedi 2004). Performability modelling together with queuing issues is especially required in order to predict the realistic performability measures and to optimise this threshold parameter for systems with various characteristics. This research topic also is covered in this thesis.

Two stage tandem networks are effectively used for analytical modelling of various communi-

cation and computer systems which have tandem system behaviour. Performance evaluation of tandem systems with feedbacks can be handled with these models. In (Chakka 1998, Gemikonakli et.al. 2006) two dimensional Markov chains are used to model these systems. Each dimension is used for random variables which represent the number of jobs in each stage. In (Gemikonakli et. al. 2006) such a model is used in order to represent a local area network and a single server system which is connected to this network. However, server failures and multi-server systems at the second stage could not be considered because of the large number of states. It is possible to divide one of the random variables in order to present the number of jobs in the system for operative and non-operative states of the server, but this will rapidly increase the number of states required for each two dimensional representation. As the number of states which are used to represent such a system increases, the existing solution methods start to suffer because of numerical cumbersomeness. It is desirable to develop an approach for two stage tandem systems, in order to ease the numerical difficulties caused by large number of states. Such an approach can be used to handle possible failures at the second stage. Furthermore, it may be possible to handle multiple servers with breakdowns and repairs. This is also a research topic addressed in this thesis.

Performability evaluation of multi-server systems with breakdowns and repairs are considered in this project. Two dimensional representation of such systems'state space is going to be used for performability modelling. The Spectral Expansion method will be employed for steady state solution of these systems.

Since analytical models are abstractions of the real world problems, predictions based on the model, should be validated against actual measurements collected from the real phenomena (Trivedi 2002). If the solution given is not exact and a certain number of assumptions have been made, benchmarking and/or simulation results should be used for validation of the analytical model (Jain 1991, Chakka 1995).

1.3 Outline of the thesis

Chapter 2 introduces the domain of the research by providing a critical review of related literature. Existing analytical modelling techniques are critically analysed and compared. Existing performance and availability modelling techniques for the systems under study are investigated and critically analysed. Solution methods for two dimensional state spaces are compared. Detailed explanation of Spectral Expansion method is given. Multi-server systems prone to breakdowns are considered in this chapter. Existing approaches are analysed and numerical results are presented for pure performance, pure availability and performability

measures. Existing studies on multi-server systems under study are critically analysed. Also existing studies on deferred repair strategies are investigated.

The approaches for performability evaluation of multi-server systems should consider reconfiguration and rebooting delays as well. These delays are expected in fault-tolerant systems which continue serving in a degraded mode in case of particular types of failures. In Chapter 3, exact performability models are developed and solved for multi-server systems with failures, repairs, reconfiguration and rebooting delays. Analytical models given in (Trivedi et. al. 1990) are extended and homogeneous multi-server systems with finite or infinite queuing capacities are considered. Also analytical models are provided for heterogeneous multi-server systems with one main and several identical nodes. Numerical results are presented, and alongside critical analysis, significant comparisons are made.

A similar approach is used for multi-server systems using farm paradigm in Chapter 4. A certain type of heterogeneous multi-server systems with one head and several identical servers are considered. Farm paradigm multi-server systems with a backup for head node are considered as well. Availability studies for such systems exist in literature (Leangsuksun et. al. 2003c, Leangsuksun et. al. 2004). However, performability models are not presented for such systems. Single head node setup has important implications on performance of such systems and analytical models for performability evaluation is important for optimisation of many parameters for both typical and highly available (HA) farm paradigm multi-server systems. Beowulf clusters are considered as case studies. Exact modelling and solution approaches are presented. In order to validate certain assumptions, simulation results are also presented comparatively.

Open networks are very widely used models especially in communication systems. Open queuing networks with relatively small number of nodes are considered in Chapter 5. Existing approaches for modelling open queuing networks are analysed, and modelling for open queuing systems with bounded queuing capacities is presented in this chapter. The IPP and joint-state modelling approaches are extended for open queuing systems with bounded queuing capacities. Since the given solution is not exact, simulation results are presented for validation of the numerical results. Also simulation results are used in order to compare the performances of two approaches, in terms of accuracy, for open queuing networks with various queuing capacities in each node and various feedback probabilities.

In Chapter 6, the homogeneous and highly available farm paradigm multi-server systems are considered. The effects of using deferred repair strategies are investigated. Analytical models are presented which represent the Markov state space of such systems. Unlike approaches given in (Temsamani and Carrasco 2002, Tang and Trivedi 2004, Sun et. al. 2005, Carrasco

2006), analytical models are presented for performability evaluation of these systems and queuing issues are taken into account in detail. Simulation results are presented for these systems as well in order to validate the assumptions done.

The analysis done on existing two dimensional models show that, if the systems have large numbers of components (servers) mathematical computations can become difficult or in some cases cumbersome. Chapter 7, represents an approach to tackle this problem in order to degrade the effects of state space explosion while the accuracy is not decreased. The new approach presented is applicable to two stage tandem network systems. The new approach makes it possible to consider breakdowns and repairs at the second stage. Also, with the new approach it is possible to consider multiple servers at the second stage. Numerical results are compared with results obtained from the Spectral Expansion method. However, for systems with more than two servers at the second stage, the Spectral Expansion method starts to suffer because of the large number of operative states. For such systems, simulations are performed and the accuracy of the new approach is validated.

Chapter 8 summarises the main contributions of the thesis and outlines some possible avenues for future studies.

Chapter 2

Literature Review

2.1 Introduction

Multi-server system models are useful to model multiprocessor systems (Trivedi 2002, Harrison and Patel 1993), nodes in communication networks, and flexible machine shops (Stecke and Kim 1989, Stecke 1992, Righter 1996, Buzacott and Shantikumar 1993, Fiems et. al. 2004) in a manufacturing environment. Such systems can be homogeneous (i.e. all processors are identical) or heterogeneous (i.e. at least one processor is different than others). Furthermore, such systems are prone to breakdowns. In systems which are prone to breakdowns and are eventually repaired, it is not possible to have a product form solution (Baskett et. al. 1975). However, the study of such systems is important because the irregularities caused by server breakdowns and repairs can have a great effect on the performance and dependability of the network. In this chapter existing models and solution techniques have been studied together with commonly used multi-server systems in order to develop exact and efficient analytical models for performance evaluation of multi-server systems.

2.2 Existing Methods for Performance Evaluation of Computer and Communication Systems

In this section, existing analytical techniques for performance and availability modelling are analysed. Pure performance and availability models are classified and each class is explained in detail. In this section a composite measure, performability is considered as well. Merits and demerits of pure performance, availability and performability modelling techniques are investigated.

2.2.1 Pure Performance Evaluation Models for Multi-server Systems

For the evaluation of multi-server systems it is possible to use pure performance evaluation techniques. These approaches assume that the system considered is available at any time. The breakdowns or failures of systems considered are not taken into account. Queuing theory, which is emphasised in queuing research has been introduced for the exact analysis of models with one server and/or one queue in 1969 (Cohen 1995). Afterwards, queuing theory has been successfully applied to computer and communication systems' performance evaluation problems. Efficient numerical algorithms have been introduced.

Queuing theory and Markov birth and death processes are commonly used in performance evaluation. Some terms and definitions are explained here. The term queue means a waiting line, and queuing theory is generally the theory of waiting lines (Deitel 1990). In queuing theory, the queues considered can be unbounded or bounded. Unbounded queues can grow as large as necessary for holding all waiting customers and the bounded queues are for holding a fixed number of waiting customers (Trivedi 2002). In order to deal with some queuing problems, a number of random variables are considered for probability distributions. Figure 2.1 shows a simple multilevel queuing system together with some of the random variables which are commonly used in performance evaluation of queuing systems.

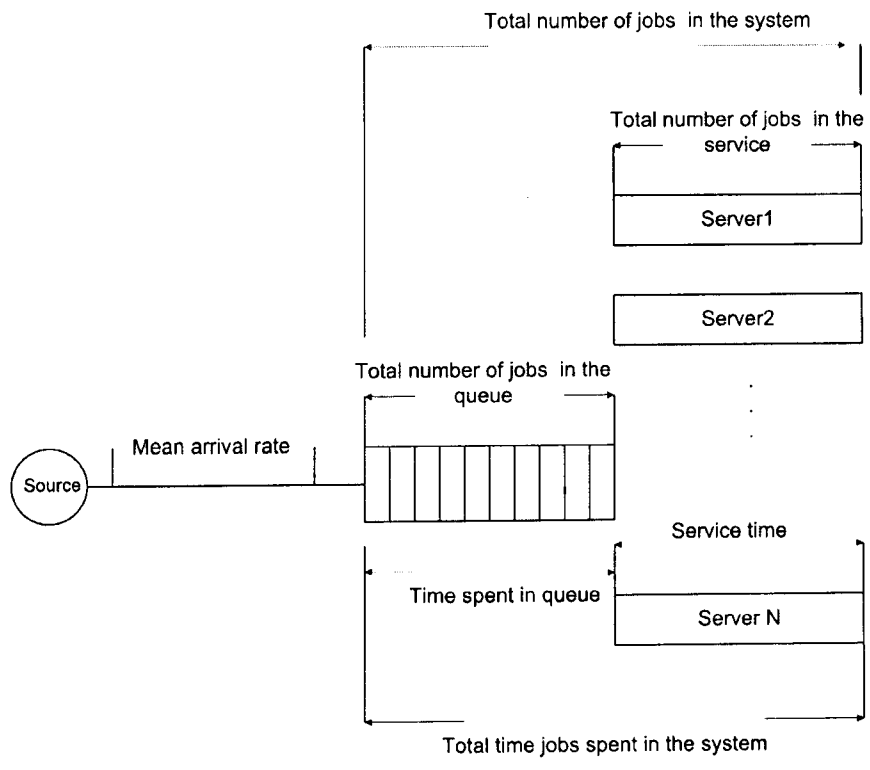

Figure 2.1: A simple, multilevel queuing system with a common queue.

In such a system jobs arrive at the system through a common queue and then they are sent to available servers. Inter-arrival and service times can be classified by using various probabilistic distributions according to the behaviour of the modelled phenomena. It is possible to group queuing networks as open and closed queuing networks. In case of open queuing networks, there are one or more sources of job arrivals and correspondingly, one or more sinks that await the jobs departing. On the other hand in a closed queuing network, jobs neither enter to nor depart from the system. In closed queuing networks arrivals and departures can be accepted as feedbacks (Deitel 1990, Trivedi 2002).

Kendall's notation is commonly used for representation of a typical queuing system. It uses six parameters to specify a queuing system. The notation, takes the form $A/S/m/B/K/SD$, where A is the type of arrival process distribution, S is the service process or the service time distribution, m is the number of servers, B is system capacity, K is the population size and SD is the service discipline of the queue (Deitel 1990). Markov process is a commonly used process for inter arrival and service time distributions (A and S in Kendall's notation) of queuing systems.

Markov processes are stochastic processes. When a Markov process evolves, future states (future state evolution or probability of all possible future states) do not depend on past states (even if there is dependence it is only on the present state). A discrete state Markov process is also called a Markov Chain.

Hence, to predict the future of a continuous-time Markov chain, it is sufficient to know the current state. It is not necessary to know the past states, how it has come to the present state, or how long it has been in the present state. In other words, the time spent in a state, which is another random variable, has a memoryless distribution (Trivedi 2002). The discrete state Markov processes where the transitions are restricted to neighbouring states, are called birth-death processes. The discrete states of these processes are usually represented by integers, and from state n it can only change to state n+1 or state n-1. A typical birth and death process is given in Figure 2.2.

In Figure 2.2 b_i and d_i are the birth and death transitions originally for state i respectively. It is possible to determine the state probabilities (P_i) by using the relations (Trivedi 2002) given as:

$$P_{i+1} = \frac{b_i}{d_{i+1}} P_i \text{ and } \sum_i P_i = 1$$

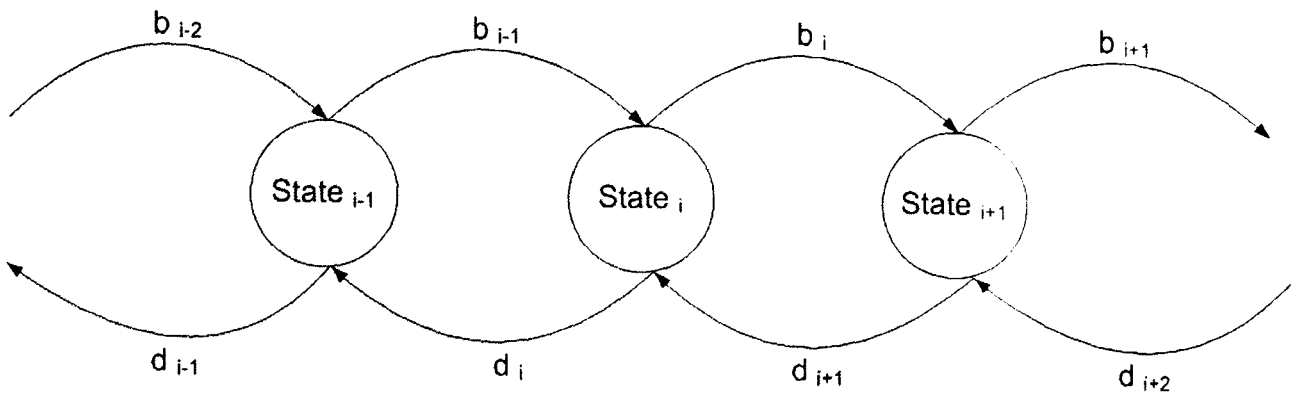

Figure 2.2: Markov birth and death processes.

Once the steady state probabilities are found, well known queuing theory formulae can be used in order to obtain various performance evaluation measures.

Theories and terminology given in Queuing theory and Markov processes are widely used in performance and availability modelling of multi-server systems. It is possible to expand existing methods for performance modelling of more complex systems. Generalisation of the single-queue solution methods to performance analysis of parallel and distributed systems is straightforward only in rare instances. Generally adaptation of queuing network results to parallel and distributed systems is only possible through certain assumptions.

In the queuing network theory, under certain assumptions it is possible to obtain a simple exact solution for the joint queue length distribution in a separable form. This form is called the product form (Trivedi 2002). In (Heidelberger and Trivedi 1982), a class of parallel processing systems where jobs subdivide in several asynchronous tasks are presented. Then the jobs of this non-product-form network are iteratively approximated by a sequence of product-form networks. Another development in product-form theory has been introduced in (Gelenbe 1991). In this study Gelenbe has considered performance analysis of resource request and allocation models with positive and negative signals (arrivals). These models have given rise to product-form networks with positive and negative customers. The main drawback of this approach is the fact that modelling of parallel and distributed systems usually does not lead to product forms.

The performance modelling of multi-server systems with multiple queues usually leads to multi-dimensional models. In particular, the analysis of Markov processes whose state space is the N-dimensional set of lattice points. A concise exposition of the method, and several applications and references, are presented in (Cohen 1995). The functional equations arising in the analysis of N-dimensional processes usually present analytic difficulties. Also in multi-dimensional modelling techniques performance measures are usually not directly applicable. Because of this, various analytic-algorithmic methods have been developed to solve multi-

dimensional queuing systems. The most important two of these methods are well known Matrix Geometric method (Neuts 1981) and Spectral Expansion (Chakka 1995) method. These methods are considered in turn. Under some conditions, if a system approaches saturation and the queue is almost full, most of the time, the performance evaluation model can become analytically intractable. Approximation techniques present an alternative to numerical methods for solving such systems. In systems with heavy traffic approximations, if the queue lengths are properly normalised, they can be approximated by Brownian motions with drift, which leads to a diffusion approximation of the system (Harrison and Williams 1987). In light traffic approximations, performance measure under study is considered as a function of the arrival rate. Derivatives of this function are computed at point zero (Reiman and Simon 1989).

The analysis of a system from the pure performance point of view tends to be optimistic. This is because it is assumed that the system under study never fails. However, in many multi-server systems, failures are expected and they have a great effect on the systems performance since in case of failures some delay stages, and/or stages where the system is in a degraded mode are possible steady states. Also when the servers in a system are prone to breakdowns and repairs, the system does not have a simple product form solution. This is because the irregularities caused by server breakdowns and repairs affect performance and dependability of the system significantly.

2.2.2 Availability Models for Multi-server Systems

The ability of a system to perform a required function or operation under certain conditions for a given time interval is called reliability. On the other hand, availability is defined as the ability of a system to be in a state to perform a function or an operation at a given instant of time, or at any instant of time within a given time interval. These two terms are closely related with each other. However, an important difference between reliability and availability is that reliability refers to failure-free operation during an interval, on the other hand, availability refers to failure-free operation at a given instant of time (Trivedi 2002).

Mathematical definition of instantaneous Availability or point availability $A(t)$ of a component which is equal to the probability that the component is properly functioning at time t is given by.

$$A(t) = R(t) + \int_0^t R(t-x)m(x)dx$$

where $R(t)$ is the probability of having no failure in interval $(0, t]$ and $m(x)$ is the repair

density. This equation shows that the system is available either if no failures occurs in interval $(0, t]$, or failure occurs but repair of the system is completed before time t (Trivedi 2002).

Then the mathematical definition of limiting A can be given as

$$A = \lim_{t \longrightarrow \infty} A(t) = \frac{MTTF}{MTTF + MTTR} \tag{2.1}$$

where $MTTF$ is the mean time to failure and $MTTR$ is the mean time to repair. The limiting availability depends only on the mean time to failure and mean time to repair, and not on the nature of the distributions of failure times and repair times.

Figure 2.3 shows the effects of MTTF and MTTR on availability measures.

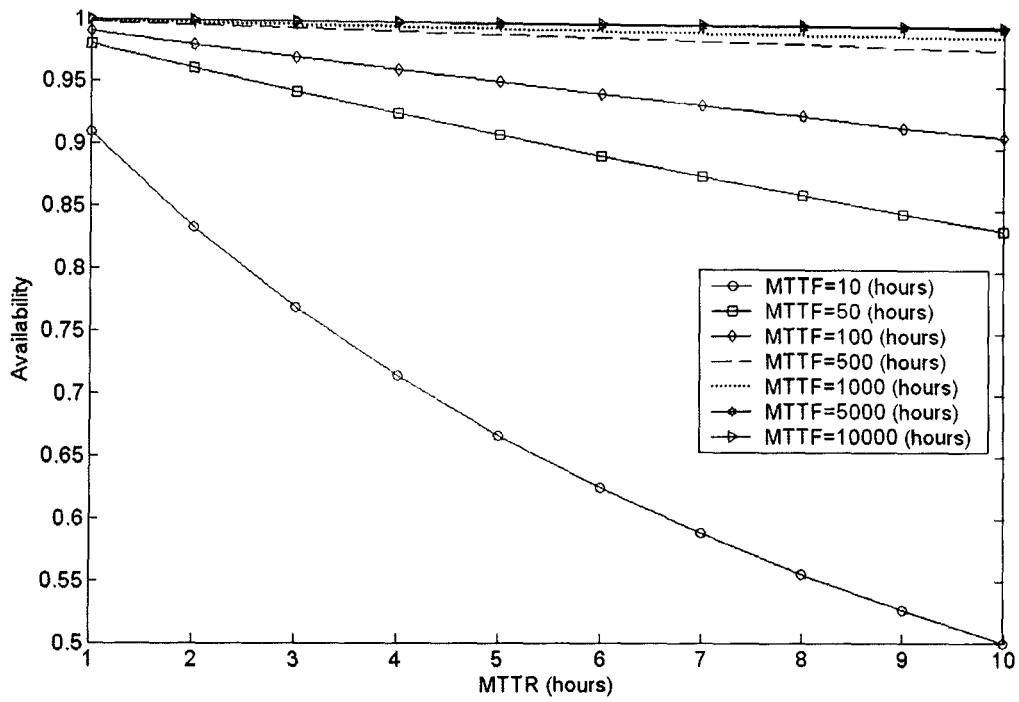

Figure 2.3: Availability measures computed

High availability refers to the ability of a system to perform its function continuously for a longer period of time than its individual components' reliabilities. This is usually achieved through fault-tolerant systems. Fault-tolerant systems usually operate continuously, and short down times during their operation can be tolerated. In these systems, both preventive and corrective maintenance can be performed to obtain the desired level of service. Trivedi groups model types used to study availability under three main titles (Trivedi 2002) as:

1. Combinatorial Model Types
2. State-space Models
3. Hierarchical Models

Combinatorial model types capture conditions that make a system fail, in terms of the structural relationships between the system components. In this approach the models are solved without generating a state space. Typical examples for these types of models are reliability block diagrams, reliability graphs, and fault trees.

A reliability block diagram (RBD) is a graphical representation of the components of a system and how they are reliability-wise connected. RBDs are effectively used to describe the interrelation between the components and to define the systems reliability characteristics (Trivedi 2002). A reliability graph contains a set of nodes and a number of directed edges between those nodes. Each edge represents a component that can fail, or a structural or dependency relationship between two components in a system configuration. The system represented by a reliability graph fails when there is no path from the source to the sink node. Each edge in the graph can be associated with a probability of failure (Trivedi 2002). Fault trees are graphic models of the pathways within a system that can lead to a foreseeable failure. The pathways interconnect contributory events and conditions, using standard logic symbols. Numerical probabilities of occurrences can be computed through the model to evaluate probability of the failures. Among the combinatorial type models, the fault tree analysis is most widely used due to its expression power, and applicability to complex systems. However, drawing a fault tree can be a cumbersome task, and requires a great amount of attention and caution to represent a system correctly (Trivedi 2002). In (Amari et. al. 2003) a new dynamic approach has been given to solve dynamic fault trees. Since combinatorial type models use fast algorithms such as the sum of disjoint products, binary decision diagrams, factoring, and series parallel composition (Trivedi 2002), they can handle systems with several hundred components. Combinatorial type models can be solved using fast algorithms only by assuming stochastic independence between system components. They assume that the failure or repair of a component does not affect other components. Also in combinatorial type models it is assumed that there are as many repair facilities as necessary.

In many practical systems dependencies have been observed among system components, and repair facilities can be restricted (Trivedi et. al. 1990, Chakka 1995, Trivedi et. al. 1996, Chakka 1998, Chakka et. al. 2002). To be able to model more complicated interactions among components, state space models can be used. A state space model is a description

of a configuration of states used as a simple model of the system under study. State space models consist of states and transitions between these states. Although non-state-space models are efficient, to specify and solve these models assumes that components of the system are independent. A failure of one component does not affect the operation of other components, and components cannot share a repair facility. State space models provide the ability to model systems with components not completely independent from each other. Markov chains, Markov reward models, stochastic reward nets and petri nets are well known state space based models.

In (Goyal et. al. 1987) probabilistic models which are developed for computer system availability are presented. Steady state probability, transient probability, and distribution of availability over a finite interval has been considered. Analytical and numerical techniques for evaluation of these probabilities are discussed by mainly focusing on Markov processes.

Also, in (Trivedi et. al. 1996) a Markov chain for computing the dependability of a multiprocessor system is presented. Steady state probabilities and availability of a multiprocessor systems is given in terms of failure rate and certain delay parameters. A method is presented in order to optimise the number of processors according to the availability of the system.

The main drawback of these type of models is the state space explosion in case of systems with large numbers of components (Trivedi 2002). It is possible to overcome the state space explosion problem by simplifying the model under study. However, in these cases, as the number of states in a model increases, the accuracy of the results decreases because of the underlying assumptions. Another way of solving this problem is using the state space models by introducing hierarchy and handling different levels independently (Trivedi 2002). These models are named as hierarchical models. In (Trivedi and Xiaomin 2002) hierarchical approach has been used for probabilistic analysis of wireless cellular networks.

Software package Symbolic Hierarchical Automated Reliability and Performance Evaluator (SHARPE) has been introduced by Trivedi (Sahner and Trivedi 1987) for specifying and analysing performance, and reliability models. SHARPE is a toolkit which provides a specification language and solution methods for commonly used model types for performance, and reliability modelling. Model types include combinatorial ones such as fault-trees, queuing networks and state-space ones such as Markov and semi-Markov (Similar to Markov but Transitions depend on the time spent in a state) reward models as well as stochastic Petri nets.

Pure availability concepts for analysis of communication networks tend to be very conservative because different levels of performance issues are not considered. A more realistic

analysis method has been introduced in (Beaudry 1978) and a conceptual framework of performability has been considered by Meyer (Meyer 1980). Beaudry has proposed (1978) a simple method for computing the distribution of performability in a Markov reward process. The combined evaluation of performance and availability is called performability. This modelling technique is useful especially for the systems which can operate in a degraded mode in case of breakdowns and failures.

2.2.3 Performability Models for Multi-server Systems

Performability models are composite models of performance and availability models. They are used to combine performance and availability/reliability concerns. If the main interest is performability for a specified operational time, then, reliability issues are considered, but if the performability for a given instant is analysed then availability issues are considered together with performance issues. Performability models specify the amount of work that will be done in a given interval where the failures and repairs affect the system.

In (Smith et. al. 1988) performability models based on Markov reward models (MRM) are presented. The behaviour of multiprocessor systems has been described as a continuous time Markov chain and associated reward rates for performance measure of each state. A systematic study has been performed on complex multiprocessor systems for computing the distribution of accumulated reward of Markov reward models. A Markov chain is presented for a heterogeneous multiprocessor system. Results mainly focus on distribution of cumulative bandwidth for each processor. Queuing issues and performability measures related to queuing theory were not investigated. Results show that instantaneous measures do not show the dynamic behaviour of the system. Importance of repair facility on performability distribution is underlined. Also results indicate that the change in failure, and repair behaviour affect the complementary distribution of accumulated reward, used for performability.

Beaudry's studies (Beaudry 1978) have been extended in (Ciardo et. al. 1990). Beaudry's method has been extended to a semi Markov reward process. A new algorithm is presented for the computation of the distribution of accumulated reward until absorption in a semi Markov reward process. The resulting semi Markov process is able to capture failures and repairs of systems and also performances of systems in degraded modes. In this study, several examples including M/M/2/K queues with breakdowns and a performability model for IBM 3081 multiprocessor systems have been considered. Distribution of system's up time is computed for given examples.

Trivedi and Benitez (1993) used performability measures to analyse the behaviour of processor arrays which are prone to permanent, transient, intermittent and near-coincident faults.

Hierarchical models are obtained to combine performance issues with availability issues. Also a Markov model is presented for the fault handling process. The software package Model Generator and Reliability Evaluator (MGRE) is used to generate the Markov models and another software tool SHARPE is used for solution of the generated models. The performance model is used to determine the reward rates coming from different levels of performances (from degraded modes in case of failures). In this study, effects of various failure types are analysed. Also, throughput measures are presented in terms of mean reward rates of systems. Queuing network models are solved only to obtain performance measures which are used as reward rates of overall Markov failure-repair model.

Performability modelling techniques given above can be efficiently used to obtain fairly accurate results for cumulative bandwidth distributions, mean reward rates (throughput) and similar measures for systems with breakdowns and repairs. However, in these studies queuing concerns such as limited queuing capacities, mean number of customers in queue, and the probability of a job to be blocked, are not analysed. These concerns are very important especially for systems which do not drop the incoming jobs but store them in a queue in case the servers are busy. Quasi birth and death processes (QBDs) are extensively used to model multi-server systems with bounded or unbounded queuing capacities.

QBDs are a special class of finite or infinite state continuous time Markov chains (CTMC) which can be characterised with a probability matrix, and combines a large degree of modelling expressiveness with efficient solution methods (Chakka 1995). Balance equations of these systems are the equations obtained from the equality of incoming and outgoing transitions to a particular state.

A process is called ergodic if it is irreducible and the corresponding balance equations of the state probabilities have a unique solution which can be normalised. Steady states do exist for such processes (Chakka 1995).

It is possible to develop an effective and accurate analytical model for performability measures of multi-server systems by using a two dimensional representation of the states of the system, in a Markovian framework. Chakka used this approach in his studies (Chakka 1995, Chakka and Mitrani 1994, Chakka and Mitrani 1996, Chakka 1998, Chakka et. al. 2002). Semi-finite or finite lattice strips of Markov states, with certain regularity and hence mathematical tractability, are used in this approach. These lattice strips of Markov states, with certain regularity, are also known as QBD and Quasi Simultaneous-Multiple-Births and Simultaneous-Multiple-Deaths (QBD-M) processes (Chakka 1995). This approach has given rise to a number of useful research contributions for performability evaluation of various queuing systems (Chakka 1995, Chakka and Mitrani 1994, Chakka and Mitrani 1996).

In this approach, the state of a system at time t is described by a pair of integer valued random variables, I(t) and J(t), specifying the server configuration (can also be termed, operative state of a multi-server system) and the number of jobs in the system, respectively. The finite or infinite lattice strips mentioned above are presented in Figure 2.4.

In general, if there are $N+1$ server configurations, represented by the values $I(t) = 0, 1, \ldots, N$, these $N+1$ configurations can be used to represent the possible operative states of the model. The model assumptions are assumed to ensure that $I(t), t \geq 0$, is an irreducible Markov process. $J(t)(\leq L)$ is the total number of jobs in the system at time t, including the one(s) in service. Then, $Z = \{[I(t), J(t)]; t \geq 0\}$ is an irreducible Markov process on a lattice strip (a QBD process), that models the system. Its state space is, $\{0, 1, \ldots, N\} \times \{0, 1, \ldots, L\}$ (where L can be finite or infinite). Once the steady state probabilities of such a system are computed, various performability measures (mean queue length, probability of jobs to be lost, mean response time, throughput etc.) can easily be driven by using well known queuing theory knowledge. A similar model was analysed for exact performability (Chakka and Mitrani 1994, Chakka 1995, Chakka et. al. 2002), for some general multi-server systems and for some repair strategies. In these studies a threshold value (M) is defined to represent the point where the transition matrices become independent from the value of $J(t)$.

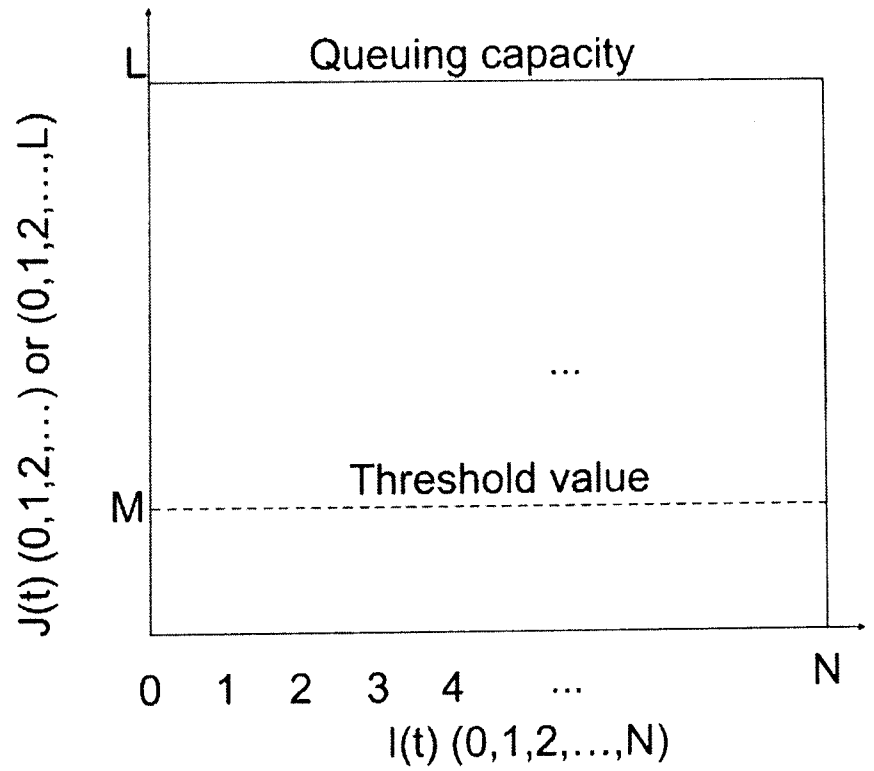

Figure 2.4: Finite or semi finite lattice strip representing the steady states.

Two dimensional representations of system states are used in this research. Random variables $I(t)$ and $J(t)$ are conceived/specified according to the characteristics of the model and the behaviour of system under study. Exponential distributions for failure and repair times are used in performability modelling of systems under study. To allow a Markov chain analysis, it is possible to assume that the time to failure of all components have an exponential distribution (Sheldon et. al. 2002, Trivedi 2002). This signifies that the distribution of the next failure time of a component does not depend on how long the component has been operating. In fault-tolerant multi-server systems the failures are mostly because of the hardware and software failures which are not due to aging. The next breakdown is the result of some suddenly appearing failure, not of gradual deterioration. There is a wealth of literature on fault-tolerant systems where components are assumed to have exponential failure and repair times because of this reason, and also for mathematical tractability (Trivedi et. al. 1990, Trivedi et. al. 1996, Sheldon et. al. 2002, Trivedi 2002, Leangsuksun et. al. 2003, Leangsuksun et. al. 2004b, Cotroneo et. al. 2005, Song et. al. 2006). The approach presented here adopts a similar approach. However alternative distributions (e.g. weibull and gamma distributions for breakdowns and lognormal distribution for repairs) have also been reported (Schroeder and Gibson 2006). For example, if failures are due to aging of components, weibull will be an appropriate distribution. Also the proposed approaches can be successfully extended to the case of phase-type distributions, at the cost of an increase in the state space. In the following section the systems under study are considered. The approaches to model these systems for performability measures are explained in detail.

2.3 Systems Under Study

This section gives the background information of systems considered for performance and availability evaluation. Existing modelling approaches for performability modelling of these systems are analysed in detail.

2.3.1 Homogeneous Multi-server Systems

In a multi-server system, the serving units chosen, usually share the same characteristics with each other. Such systems are widely used especially in cluster computing and multiprocessor computing facilities such as processor arrays. Queuing theory can be effectively used for performance evaluation of such systems but in case of server breakdowns, the performance measures show great irregularities.

Chakka has considered a homogeneous multiprocessor system with N identical processors,

serving a common queue where the processors are prone to breakdowns (Chakka 1995, Chakka 1998). The system has been analysed for exact performability. Queues with bounded and unbounded queuing capacities have been considered for homogeneous jobs. In this study, single, independent repairs of processors as well as multiple, simultaneous repairs are considered. In such a system when the server is operative, each server serves one job at a time, and each job can be assigned to one operative server at a time. In case of interruption of service because of failures, the interrupted job is either resumed, or repeated with re-sampling after the repair of the server. In (Chakka 1995, Chakka 1998) failure, repair and service times are taken as exponentially distributed. Semi finite and finite lattice strips were used for two dimensional modelling of the systems, and Spectral Expansion method has been used for steady state solution of the two dimensional model. Measures such as the mean number of jobs in the system, 95^{th} percentiles, and probability of jobs to be lost were computed for comparison of various systems. A simplified CTMC for a system with K processors is shown in Figure 2.5.

In Figure 2.5 the breakdown and repair rates are given by ξ and η respectively (corresponding to exponentially distributed time-to-fail and repair times). Chakka's model gives an exact solution for homogeneous multi-server systems. The model can be used to obtain exact performability measures for systems with bounded and unbounded queuing capacities.

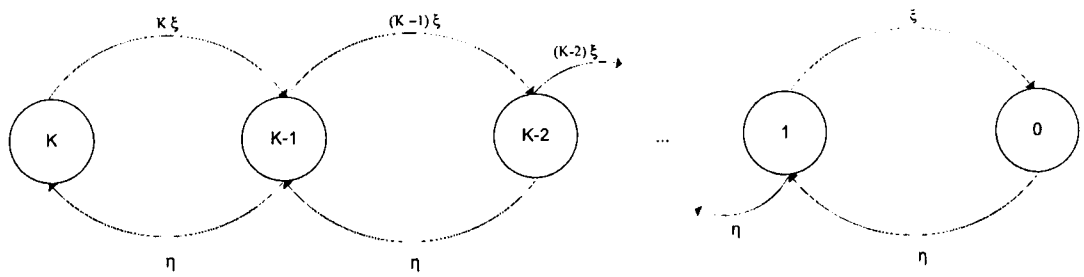

Figure 2.5: CTMC for a homogeneous system with repairs and breakdowns.

It is possible to compare pure performance models (no breakdowns) and performability models in order to show the effects of breakdowns and repairs. The well known queuing theory formulae (Little's Theorem) are used for computations of pure performance model (figure 2.2). The arrivals follow poisson distribution with mean arrival rate σ, W is the mean time customers spend in the system, W_q is the mean queuing time and $E(s)$ is the exponentially distributed mean service time.

$$MQL = \sigma W,\ W = W_q + E(s)$$

The Spectral Expansion method is used for obtaining performability results. The MQL values are computed as a function of σ. Computations are performed for various, ξ (1/hours)and η (1/hours) values. The mean service rate is taken as $\mu = 2$ jobs/hour. For simplicity, a homogeneous system with four servers is considered. The results are illustrated in figure 2.6. Figure 2.6 shows that the breakdown, repair behaviour of multi-server systems have significant effects on system performance especially if high failure rates are expected.

The model assumes that a failed processor can be mapped out of the system and a repaired processor can be re-admitted to the system without causing any delays. However in practice, when multi-server systems are used, some delay is encountered when a failed processor is being mapped out of the system or a repaired processor is being admitted into the system. These delays can be classified as reconfiguration and rebooting delays.

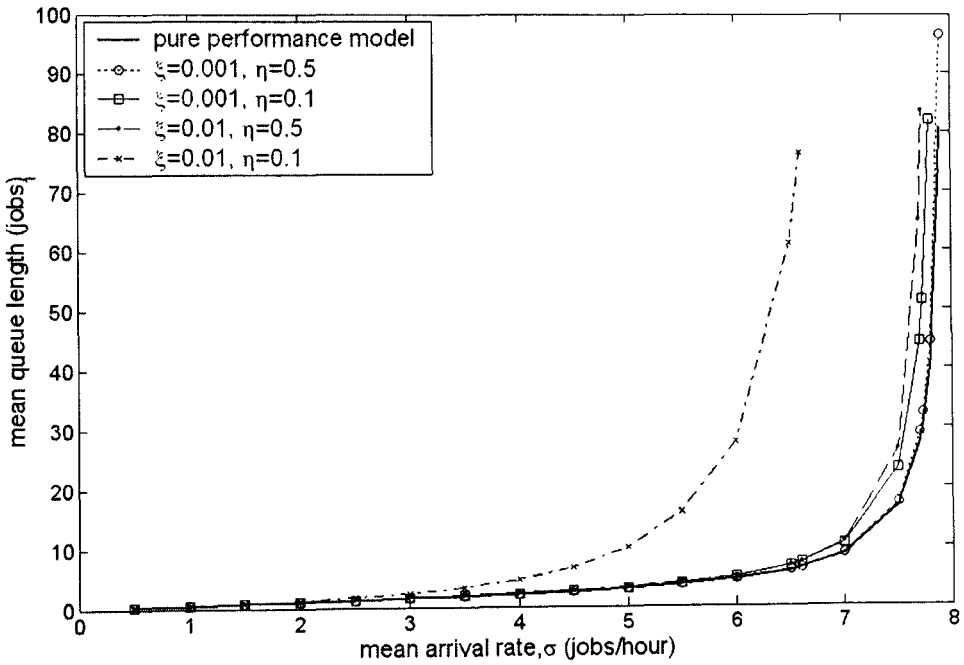

Figure 2.6: Comparison between pure performance and performability computations.

An approximate performance modelling of systems with rebooting and reconfiguration delays was proposed in (Trivedi et. al. 1990, Trivedi et. al. 1996). A model is developed to show the effects of the reconfiguration and rebooting delays. A CTMC of a homogeneous multi-server system with reconfiguration and rebooting delays is presented using a Markov reward model to analyse performance and availability. Effects of increasing the number of processors on the effectiveness of a multiprocessor system are analysed. Performance, availability, and several combined measures of performance and availability measures are considered. The

total loss probability of a task is defined as the sum of rejection probability, due to the system being down or full. Results show that the system's performance is likely to improve with an increased number of processors but systems availability does not always increase as the number of processors increases.

In (Chakka et. al. 2002) a similar model with reconfiguration and rebooting delays is given for performance evaluation of heterogeneous multi-server systems. Also a comparative study and a framework for performability analysis of heterogeneous multi-server systems with various repair strategies is presented. However, this work presents only a preliminary theoretical framework without any numerical results for bounded or unbounded systems.

2.3.2 Multi-server Systems with Farm Paradigm Architecture

One of the major problems in multi-server systems with parallel computation facilities is the ease of use, and reuse. In order to use the high performance ability of parallel computing systems in a more user convenient way there are various parallel programming paradigms used in the literature (Wagner et. al. 1997). Software components which use these paradigms hide the complex details of distributed system architecture and allow the programmer to focus on the computation rather than the parallelisation and the coordination of the multiple processes. One of the most commonly used parallel computing paradigms is the processor farm paradigm. The term processor farm paradigm is used to describe the job handling strategy used by the overall multi-server system. The idea behind this sort of parallel processing is to provide supercomputer performance, and cut the time required to perform complex computations, by sharing the necessary work among the multi-server system's nodes. Farm paradigm is commonly used in various cluster systems such as Beowulf, and Myrinet based clusters.

Although farm paradigm provides flexibility, and a very effective strategy for parallelising an application, performance varies greatly, and depends not only on the application, but underlying machine as well (Wagner et. al. 1997, Adams and Vos 2002). To effectively utilize such a system, it is important to identify the key parameters which affect performance and availability. Also the behaviour of the system should be studied in detail.

One possible architecture to implement farm paradigm is to have one main and several identical processors serving the same stream of incoming jobs (Adams and Vos 2002). In this architecture the identical processors can continue serving even if the main processor fails. The main processor usually has greater service power since it is responsible for the distribution of jobs. In case of the failure of the main processor, identical nodes continue to serve in a degraded mode (Adams and Vos 2002). Clearly these systems are prone to

breakdowns as well. If a processor fails and system continues serving in a degraded mode, reconfiguration and rebooting delays are expected.

Another and more common architecture is to have a head node and several identical computing nodes. In this architecture the identical nodes cannot compute in case of head node failures. This is because the job distribution and scheduling facility runs only on the head processor and job distribution is not possible in case of head processor failures. This architecture is commonly used especially in Beowulf clusters.

A Beowulf cluster is a multiprocessor system built from commodity off-the-shelf personal computers connected via a dedicated network and free open-source software (Brim et. al. 2001, Adams and Vos 2002, Leangsuksun et. al. 2004). The Beowulf-style clusters became very popular since they allow the computational power of multiprocessor systems to be available for parallel or simultaneous processing at much lower costs (Wagner et. al. 1997, Adams and Vos 2002). Until recently to have this computational power on commercial supercomputers such as Cray, has been prohibitively expensive and not many organisations can afford to have them.

A typical Beowulf cluster has two types of nodes. These are a head node server and multiple identical client nodes. The server or head node is responsible for serving user requests (jobs) and distributing them to clients via scheduling/queuing software. Clients or identical nodes are only dedicated to computation and they do not have any job distribution or scheduling facility. Since the head node is responsible for organisation and distribution of jobs, the clients cannot compute if the head node is not operative (Brim et. al. 2001, Leangsuksun et. al. 2004, Leangsuksun et. al. 2005). Because of this single head node setup, clusters are vulnerable, as the head node represents a single point of failure affecting availability of the cluster. Furthermore, the head node represents a single point of control (Leangsuksun et. al. 2005). This has a great effect on performability measures since the system is completely down in case of head processor failures.

There are a number of techniques available to solve the bottleneck problem caused by this single head node architecture. The standby replacement approach is an economical and efficient way of achieving redundancy at reasonable cost for critical processors in the control hierarchy. There are mainly two different standby techniques. These are cold and hot standby techniques.

Cold standby technique can be defined as planning and configuring a backup policy for an unplanned but expected down time. The effective down time should not be more than the time required to recover the systems and restarting them. The recovery time should be measurable and consistent. The topology for this technique can be a configured backup

processor and a plan to recover the services. The operator follows the plan to bring the services back up in a pre-defined time frame.

On the other hand, hot standby technique allows for a minimal downtime during an outage. The critical applications which can risk a minimal down time use this technique. The topology for hot standby clusters is to have a configured backup processor started up and set in a standby mode waiting for a flip of a switch which will direct the new traffic to the backup processor and will allow the older traffic in the faulty cluster to fade away. The switch can be automatic or operator controlled. HA-OSCAR is the first highly available (HA) Beowulf cluster which provides high availability and a critical failure prediction capability by using hot standby technique (Leangsuksun et. al. 2004).

Performability of cluster systems are considered in (Nagaraja et. al. 2005). A two-phase methodology is proposed for evaluating the performability of cluster-based Internet services. In the first phase, evaluators use a fault-injection infrastructure to characterize the service's behavior in the presence of failures. In the second phase, evaluators use an analytical model to combine an expected fault load with measurements from the first phase to assess the service's performability. To demonstrate the methodology, the performance, availability, and performability of three soft state maintenance strategies in the context of a multitier bookstore service are considered.

The cluster systems with farm paradigm are not considered for performability studies till now, though this is a very important research topic. Pure performance studies and availability studies are reported separately. In (Pratt 1998) benchmarking results are used together with Godiva software instrumentation tool for comparing the performances of CRAY and Beowulf cluster systems. The performance evaluation of a Beowulf-class system in a typical computational fluid dynamics scenario is considered in (Garcia et. al. 2002). Different configurations of a PC Cluster are compared by using two multi-grid codes as target applications. Analytical modelling for performance evaluation of various cluster systems is considered in (Mohamed et. al. 2003). Cluster systems using farm paradigm architecture are not considered.

Recently highly available cluster computing has gained much attention, especially for mission critical and enterprise applications (Lengsung et. al. 2003). Performance analysis has been relatively matured enough to estimate the total system performance. Also system dependability analysis is considered in various studies (Leangsuksun et. al. 2003, Leangsuksun et. al. 2003b, Leangsuksun et. al. 2004).

An initial approach, where only the two-tier computing system is considered can be found in (Leangsuksun et. al. 2003c). In this study, only the hardware failures for each entity

is considered. Dependencies between components are ignored. A well-integrated tool that automatically transforms Unified Modeling Language (UML) model and calculates reliability in a single step is developed. The HA-OSCAR system models are used to predict the dependability of the system by a Petri net-based model, Stochastic Reward Net in (Leangsuksun et. al. 2003). The high availability features are evaluated and compared with dependability requirement of the system. The failure rates are assumed to be exponentially distributed. The availability of Highly Reliable Linux high performance clusters (HPC) with Self-awareness Approach is evaluated in (Leangsuksun et. al. 2004). This time Stochastic Petri Net Package (SPNP) is utilised to build and solve the Stochastic Reward Nets (SRN) model. Dependability Analysis of highly available Beowulf clusters is also considered in (Leangsuksun et. al. 2003b). Continuous Time Markov Chain (CTMC) model formalism is used for dependability analysis of computer systems since CTMC can easily handle many of the interdependencies and dynamic relationships among the system components.

Beowulf clusters are very widely used for parallel processing all over the world. These systems can continue serving in a degraded mode while the failure is not at the head node. For such systems the combined evaluation of performance and availability is the most realistic approach. However, performance modelling for Beowulf clusters with one head and several identical processors has not been considered together with breakdowns and repairs, so far.

2.3.3 Tandem Systems

Multi stage tandem networks have been considered in (Chakka 1995, Chakka 1998). Two dimensional performability models for systems with bounded and unbounded queuing capacities are presented. These models can be used in performance modelling of various communication and computer systems which have tandem system behaviour, with or without feedback.

A model for performance modelling of a two stage tandem network is presented in (Chakka 1998). The model considered is shown in Figure 2.7

Two dimensional modelling technique shown in Figure 2.4 has been used to represent this system. Instead of operative states of the system, the number of jobs in the bounded queue is presented as values of random variable $I(t)$. The Spectral Expansion method was used for computation of steady state probabilities of a system with finite queuing capacity. Numerical results are presented for number of customers in each queue, and blocking probability of the queue with bounded queuing capacity.

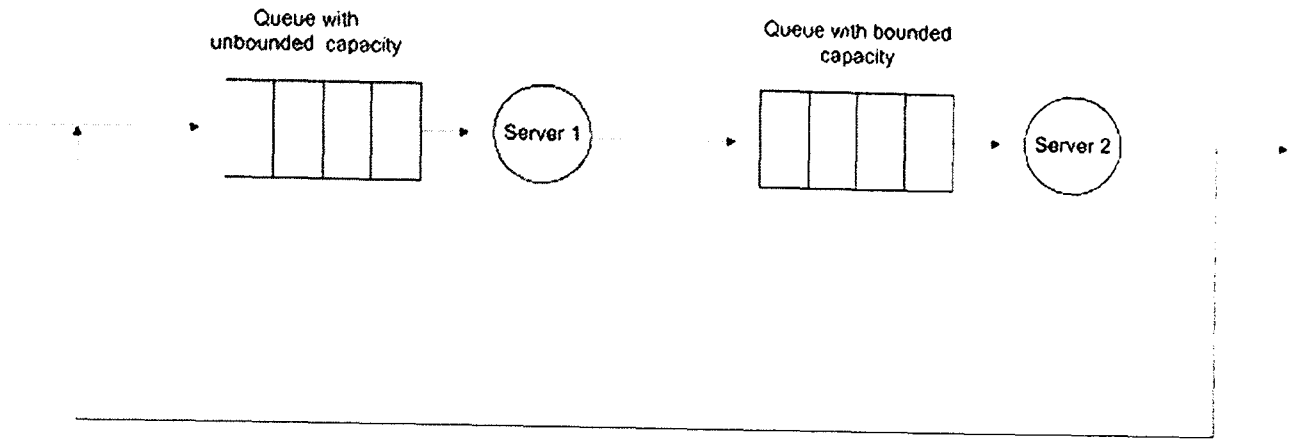

Figure 2.7: A two stage tandem network with feedback.

It is difficult to use this model for systems with relatively large queuing capacities. This is mainly because the random variable $I(t)$ holds the number of customers in the bounded system. In case of queues with relatively large capacities the state explosion problem is expected. Also in this study the measures computed are pure performance measures since breakdowns and repairs of the servers are not taken into account. Systems with multiple servers in the second stage are not considered either. Open network models with various arrival and departure process models (e.g. IPP, and MMPP) can be used to handle multistage tandem networks with breakdowns and repairs. However, state explosion problem is inherent to all such models.

2.3.4 Open Queuing Networks with Breakdowns and Repairs

Open queuing networks are useful models for computer networks, and modern communication systems with inter-connected nodes. It is not possible to use the product form solution approach, since many practical systems have servers which are prone to breakdowns (Chakka 1995, Chakka 1996, Chakka et. al. 2000), thus giving rise to non product form networks.

When a single node in a network of nodes is considered, it is possible to take the superposition of Poisson external arrivals and internal arrivals fed by other nodes, by considering it as a single stream with bursty arrivals. In (Ny, and B. Sericola 2002, Dudin et. al. 2005, Klimenok et. al. 2005) various approaches are given to model systems with bursty arrivals. Batch Markov arrival process (BMAP) is effectively used to represent the superposition of identical MMPPs in (Ny and Sericola 2002). In (Dudin et. al. 2005) the BMAP/G/1/N system with complete admission and complete rejection disciplines is investigated. However, in these approaches unreliable servers with breakdowns are not taken into account. In (Li et. al. 2006) the batch Markov arrival process (BMAP) is used as a mathematical model for describing bursty traffic in a single server system with a server subject to breakdowns and

repairs. Reliability characteristics such as the stationary availability, the stationary failure frequency and the reliability function are considered as well as queueing characteristics such as the stationary queue length and the busy period.

Chakka used several other engineering techniques together with the Spectral Expansion method in order to find an acceptable approximate solution for performability analysis of open networks with Poisson arrivals, unbounded buffers, breakdowns and repairs (Chakka 1995, Chakka et. al. 2000). He analysed an open network with a number of nodes where each node contains a server and an unbounded first in first out (FIFO) queue. In such a system, the total flow of jobs can be external or the superposition of flows from other nodes (if feedback is an option).

Two approaches are presented in (Chakka 1995 and, Chakka et. al. 2000) for performability analysis of open queuing networks with unbounded queuing capacities. These methods are called the MMPP model and the Joint-state model approaches. The servers are prone to breakdowns and once they become broken they are set into repair process immediately. After the completion of the repair process, servers become operative again. The service, breakdown and repair times of the servers are exponentially distributed. Also the inter arrival times of incoming jobs are taken as exponentially distributed. The queuing capacity of each node is assumed to be unbounded. After a job is serviced at node k, it is routed to node l with a probability of $q_{k,l}$. It is assumed that $q_{k,k} = 0$ $(k = 1, 2, \ldots, K)$. The probability of a job to leave the system after being served in node k is $q_{k,K+1} = 1 - \sum_{l=1}^{K} q_{k,l}$. For a system with K nodes Q is the routing probability matrix of size $K \times (K+1)$ such that $Q(k,l) = q_{k,l}$ $(k, l = 1, 2, \ldots, K)$.

In (Chakka 1995, Chakka et. al. 2000) each node in the system has been considered as a MMPP/M/1 model. Interrupted Poisson Process (IPP) has been used for departure modelling of these nodes. A single node system with MMPP arrivals is analysed in (Chakka 1995). The number of phases of the MMPP arrival process is taken as T. That means; the associated Markov process has T states. A generator matrix which represents the associated continuous time, irreducible, homogeneous Markov process of the MMPP, and a diagonal matrix of size $T \times T$ whose i^{th} diagonal element is the arrival rate of the MMPP in phase i is used to characterize the arrival process.

Random variables $I(t)$ and $J(t)$ are used again for two dimensional modelling. The first T states of $I(t)$ represent the breakdown states of the server, and remaining T states represent the operative states with respective arrival phases of the MMPP. The system is solved by using the Spectral Expansion method.

In queuing networks, the departed jobs may be fed to other relevant queues, as arrivals for further servicing. Hence departure process modelling is an important issue for analysing the open networks. Chakka has used IPP for departure process modelling. IPP can be defined as a two phase MMPP where flow rate in one of these phases is equal to zero (Fischer and Meier Hellstern 1993). The parameters of the departure process can be defined as (ν, α, β). Here υ is the departure rate in phase one. In phase two the departure rate is zero. Transition rate from phase one to phase two is α, and from phase two to phase one is β . Computation of these parameters is done by computing the moments of the inter-departure times, from the parameters of the system, and working out the parameters from these computed moments (Chakka et. al. 2000). These parameters are mathematically defined as follows:

$$\alpha = \frac{9(d_2 - 2d_1^2)^3}{(6d_1^3 - 6d_1 d_2 + d_3)(2d_1 d_3 - 3d_2^2)}; \; \beta = \frac{3(d_2 - 2d_1^2)}{6d_1^3 - 6d_1 d_2 + d_3}; \; \nu = \frac{2(6d_1^3 - 6d_1 d_2 + d_3)}{2d_1 d_3 - 3d_2^2}$$

where d_1, d_2, d_3 are the first three moments of the random variable, representing the inter arrival times of the departure process.

An iterative procedure is given in (Chakka 1995, Chakka et. al. 2000) to solve the model given to represent open queuing networks with breakdowns, repairs and infinite queuing capacity, for different performance evaluation parameters. To be able to use the IPP model for a departure process, some assumptions have been made. Let S_k be the set of nodes which the departures from node k are routed and G_k be the set of nodes which node k receives jobs from. Then, the split process of the departure IPP from node k, entering node l is also approximated by an IPP with parameters $\nu_{k,l}$, $\alpha_{k,l}$, and $\beta_{k,l}$. The internal arrival $\text{IPP}_{k,l}$ and the external Poisson to node k are assumed to be Independent, for each node k. With these assumptions, node k receives one independent IPP from each node belonging to set G_k plus its independent external Poisson arrivals. The superposition of all these streams (assumed to be independent) entering node k, is an MMPP with 2^{g_k} phases. Where g_k is the number of independent IPPs node k receives from the nodes belonging to G_k. Then, each node can be considered as an MMPP/M/1 model and solved by following the procedure given below.

Probabilistic splitting of IPP and constructing MMPP are very important issues used in iterative procedures used. Detailed information about these processes can be found in (Fischer and Meier Hellstern 1993).

The iterative solution given for the MMPP model (Chakka 1995, Chakka et. al. 2000) can be summarised as follows:

1. In the first step the total average arrival rates to all nodes are computed. Each node is solved as an MMPP/M/1 queue with breakdowns and repairs by using the computed

average arrival rates and the Spectral Expansion method. In this step, MMPP_k are considered as Poisson processes with average arrival rates. Also in this step mean queue lengths MQL_k and departure process parameters are computed.

2. In the second step the IPP_k computed in previous iteration are used to re-construct the MMPP_k. Each node is again solved as a MMPP/M/1 queue by using the Spectral Expansion method. New MQL_k values are calculated. In this step the variable ϵ is defined as $\epsilon = \sum_{k=1}^{K} |MQL_k - MQL_k^{(old)}|$.

3. If ϵ, computed in the previous step, is greater than a specified tolerance value then the procedure goes back to step 2.

4. If ϵ, computed in the previous step, is less than the pre-specified tolerance value then all required steady state performance measures are computed with the most recent parameter values.

Unlike the MMPP model, the joint state approach uses a binary random variable $O_k(t)$ which is equal to zero in case the server k is broken and one in case the server k is operative. Then, the joint-state of all the servers for an open network with K nodes is an ordered K-bit binary random variable, $(O_1, O_2, \ldots, O_K)$. The total number of states are 2^K.

Since the total job arrival and total job departure processes within any given joint-state considered are Poisson, it is possible to use the algorithm given in (Chakka 1995, Chakka 2000) together with Spectral Expansion solution and analyse open queuing networks for performability measures. In this approach each node k is again taken as Markov-modulated queue or an MMPP/M/1 correlated queue but this time the job arrival process has 2^K phases, and these phases are synonymous with the joint-states of the servers. In phase i (which is the joint-state i), the arrival rate at node k is denoted by $\hat{\sigma}_{k,j}$ and the service rate is denoted by $\mu_{k,j}$.

The random variable $I(t)$ is used to represent the operative state of the system by using the K-bit binary random variable, $(O_1, O_2, \ldots, O_K)$. The transition matrices are defined accordingly and Spectral Expansion method is used to solve the system for state probabilities. Once the steady state probabilities $P_{i,j}$ obtained the total rate of job departure process $\hat{\nu}_{k,i}$ can be expressed as:

$$\hat{\nu}_{k,i} = \mu_{k,i}\left(1 - \frac{P_{i,0}}{\sum_{j=0}^{\infty} P_{i,j}}\right); i = 1, 2, \ldots, 2^K - 1 \qquad (2.2)$$

and $\hat{\sigma}_{k,j}$ can be computed as:

$$\hat{\sigma}_{k,j} = \sum_{l=1}^{K} \hat{\nu}_{l,i} q_{l,k} + \sigma_k; i = 1, 2, \ldots, 2^K - 1 \tag{2.3}$$

Another iterative solution is presented in (Chakka 1995, Chakka et. al. 2000) in order to compute the mean queue length values of each node by using the joint state model approach. The iterative solution can be summarised as follows:

1. The iteration starts with $\hat{\sigma}_{k,j} = \hat{\sigma}_k$ just like the iterative solution given for the MMPP model. Then, each node is solved by using the given transition matrices and Spectral Expansion method. The mean queue lengths MQL_k and necessary state probabilities are computed. For each node $\hat{\nu}_{k,i}$ are computed by following equation 2.2.

2. For each node, the previously computed $\hat{\nu}_{k,i}$ are used to compute new $\hat{\sigma}_{k,j}$ by following equation 2.3. Then, each node is again solved with new parameters. MQL_k values and necessary state probabilities are computed. For each node $\hat{\nu}_{k,i}$ are computed again. In this step the variable ϵ is defined as $\epsilon = \sum_{k=1}^{K} |MQL_k - MQL_k^{(old)}|$

3. If ϵ, computed in the previous step, is greater than a specified tolerance value then the procedure goes back to step 2 otherwise it goes to step 4.

4. Iteration stops and all required steady state performability measures are computed with the most recent parameters.

Because of the assumptions made, the solutions in (Chakka 1995, Chakka et. al. 2000) are approximate solutions. Since the solution is not exact, a comparative study with simulation results is also presented. A study is presented for a three stage tandem network by modifying the routing probability matrix Q accordingly. The main drawback of this work is, Open networks which consist of nodes with finite queuing capacities are not considered. It is worth noting that many or most of the practical systems have finite queuing capacities at nodes. Therefore, it is important to consider finite queuing capacities in order to make the study more suitable for practical system applications.

2.3.5 Multi-server systems with Deferred Repairs

Deferred repair strategies are effectively used by many modern, practical, fault-tolerant multi-server systems. This repair strategy is a practice of allowing the system to serve in a degraded mode by postponing prudent but non-essential repairs. Systems with traditional repair

strategies request for a repair as soon as a component fails in order to reduce the risk of system failure caused by the accumulation of component failures. However, as systems become more complicated and the number of components in systems increases, this strategy may cause high service cost in terms of time and labour (Sun et. al. 2005, Carrasco 2006). Using deferred repairs is an alternative repair strategy which leave the failed components in the system without repair until the number of failed components exceeds a predefined threshold (Bossen et. al. 2002, Tang and Trivedi 2004, Sun et. al. 2005, Carrasco 2006).

When the systems using the traditional repair-upon-failure strategy and the systems using deferred repairs are compared, the traditional repair-upon-failure strategy is expected to have higher availability measures. However, the deferred repairs strategy can reduce the cost, particularly for the systems which the replacement of failed components is an expensive procedure (e.g. the system is at a remote location). Also, since the cost of components such as memory chips and disks, as well as the new high-end server and storage facilities are decreasing, systems can afford to tolerate multiple component failures without repairing failed components before the number of failed components reaches a threshold or a scheduled maintenance action occurs (Tang and Trivedi 2004). This design approach may reduce both downtime and service cost.

In the development of highly available computer server, storage, and networking systems, system designers perform reliability, availability, and serviceability modelling to assess operational availability, performability, and service cost achievable by the architectures under consideration in order to optimise various designs. Performability evaluation of multi-server systems with deferred repair strategies is very important since the systems are fault-tolerant and deferred repairs affect the system significantly in terms of availability.

In (Temsamani and Carrasco 2002) a new numerical method, called split regenerative randomisation, is developed for the transient analysis of a class of CTMC models with a particular but quite general structure. CTMCs are used to present reliability models of repairable fault-tolerant systems with deferred repairs.

In (Carrasco 2006) a new importance sampling scheme for the efficient simulation of CTMC models of fault-tolerant systems with deferred repair is presented. Numerical results are presented which show that using the new importance sampling schemes makes it possible to achieve highly accurate estimates in more reasonable CPU times.

An example model is analysed for systems incorporating deferred repairs in (Tang and Trivedi 2004). In order to quantify their reliability, availability, and serviceability behaviour in the useful life time, interval availability metrics are used for systems with deferred repair

strategies. A hierarchical modelling approach is followed in order to obtain performability measures. The approach has been implemented in a Sun internal web-based reliability availability and serviceability (RAS) architecture modelling and analysis tool, RAScad. Results presented have been independently verified by solving models using another software package, SHARPE.

An approach is presented to optimise repair strategy for multi-processor systems with deferred repairs in (Sun et. al. 2005). Tools such as RAScad and SHARPE are used to compute the combined interval availability, interval performability, and interval repair cost. Numerical results are presented for availability measures such as annual downtime, and annual performance loss. The examples considered indicated that by varying certain system parameters it is possible to perform optimisation analysis and identify the best deferred repair service strategy that requires much lower service cost than the service-upon-failure strategy.

In these studies analytical modelling for evaluating the performability of multi-server queuing systems with deferred repairs is not considered. Queuing characteristics such as bounded and unbounded queuing capacities and the MQL of various systems have not been considered. This is another important research item which is tackled in this thesis.

2.4 Existing Solution Methods for Two Dimensional State Space

Once the quasi-birth-death Markov model is obtained, it can be solved for its steady state probabilities by using any of the several different methods. The best known ones of these methods are Seelen's method, Matrix-geometric solution, Gauss-Seidel iterative method, and the Spectral Expansion method.

When two dimensional representation is considered in case of queueing, it is possible to define steady state probabilities using row vectors v_j as $v_j = (P_{0,j}, P_{1,j}, \ldots, P_{N,j})j = 0, 1, \ldots, L$ where L represents the queue capacity and L can be finite or infinite. This notation shows that elements of vector v_j are the steady state probabilities where there are j jobs in the system.

2.4.1 Discussions on Existing Solution Methods

In this section existing solution methods are briefly discussed. The decision of using the Spectral Expansion method in our computations is also justified.

Seleen's method gives an approximate solution for QBD and QBD-M systems. The Markov Chain is first truncated to a finite state, which is an approximation of the original process. Then, it is used together with a dynamically adjusted relaxation factor in an efficient iterative solution algorithm (Seelen 1986). In each iteration the algorithm computes one of the state probabilities. That means the computation time required may be proportional to the size of the buffer. However, it is stated that, the number of iterations needed for accurate results, does not depend on the buffer size (Seelen 1986). The dependency of computation time on the number of servers is also not stated. The use of an appropriate value for the relaxation parameter is important to obtain the most accurate results. However, this method does not define any solutions to determine the value of the relaxation parameter (Seelen 1986, Chakka 1995).

Block Gauss-Seidel iterative method developed in 1989 is used for solving many kinds of simultaneous equations iteratively. In this method, the infinite-state problem is reduced to a linear equation involving vector generating functions and some unknown probabilities. The computations appear to be serial. Since each component of the new iterate depends upon all previously computed components, the updates cannot be done simultaneously. There is no mention of the dependence of the number of iterations to buffer size or the number of servers (Horton and Leutenegger 1994, Dayar 1998). Seelen's method and block Gauss-Seidel iterative method are not as popular as the other two methods because of the drawbacks mentioned above.

In matrix-geometric solution method first a non-linear matrix equation is formed from the system parameters. Then the minimal non-negative solution R is computed by using an iterative algorithm. This method has probabilistic interpretation for each step of the computations (Neuts 1981). The main drawback of this method is that the number of iterations for computing R cannot be predetermined and there is a great computational requirement to obtain R. Another observation is that; in matrix-geometric method, for some values of certain parameters (for example, near system saturation points in ergodic queues, heavily loaded systems), the computational requirements are uncertain and relatively large.

Spectral Expansion is an exact solution technique for the steady state analysis of certain two dimensional Markov processeses in semi-infinite or finite lattice strips. These processes are mostly arising in performance and dependability problems of computing and communication systems. In Spectral Expansion method, first, the necessary matrices are computed by following the given algorithm. Then, eigenvectors and eigenvalues are computed to obtain a system of linear equations. It is possible to solve a system of linear equations with a variety of methods. One example is LU Decomposition method. In his studies, Chakka has applied

the Spectral Expansion method to several non-trivial and complicated modelling problems, occurring in computer and communication systems (Chakka 1995, Chakka and Mitrani 1994, Chakka and Mitrani 1996, Chakka 1998). Performance measures are evaluated and optimisation issues are addressed. Also a comparative study is performed to show that the Spectral Expansion algorithm has an edge over the matrix-geometric method in computational efficiency, accuracy and ease of use. In matrix-geometric method, computation of R has a much greater computational requirement than computational requirement for eigenvalues and eigenvectors in Spectral Expansion solution. However, in Spectral Expansion method accurate eigenvalues and eigenvectors are necessary since the performance measures can be quite sensitive to these. There are various libraries that provide routines to compute very sensitive eigenvalues and eigenvectors for given matrices. One of these libraries is the Numerical Algorithms Group (NAG) library. Matrix-geometric solution method and Spectral Expansion methods are the most commonly used solution techniques for quasi-birth-death Markov models (Chakka 1995). Matrix-geometric method is developed by Neuts (Neuts 1981) and the Spectral Expansion method has been applied to the steady state solution of quasi-birth-death Markov models by Mitrani and Chakka (Chakka 1995, Chakka and Mitrani 1994, Mitrani and Chakka 1995). These two methods have been critically analysed and performances of two methods have been compared (Mitrani and Chakka 1995, Haverkort and Ost 1997, Tran and Do 1999). In these studies it is stated that the Spectral Expansion method is a better solution method especially when more heavily loaded systems are studied, and when batch arrivals (or departures) are included in the model.

In this research Spectral Expansion method has extensively been used for steady state solution of QBD Markov models, developed for various kind of multi-server systems. The following section briefly explains algorithm that the Spectral Expansion method follows.

2.4.2 The Spectral Expansion Method

Spectral Expansion is a solution technique which is useful in performance and dependability modelling of discrete event systems. It solves Markov models of a certain kind that arise in several practical system models. The reported applications and results include, performability modelling of several types of multiprocessors, multi-task execution models and networks of queues with unreliable servers (Chakka 1995, Chakka and Mitrani 1994, Chakka and Mitrani 1996, Chakka 1998) .

In this approach, matrix A is defined as the matrix of instantaneous transition rates from state (i, j) to state (k, j) with zeros on the main diagonal. These are the purely lateral transitions of the model Z. Matrices B and C are transition matrices for one-step upward

and one-step downward transitions respectively. The transition rate matrices do not depend on j for $j \geq M$, where M is a threshold having an integer value. The process Z evolves with the following instantaneous transitions:

$A_j(i,k)$: Purely lateral transition rate, from state (i,j) to state (k,j), $(i = 0,1,\ldots,N; k = 0,1,\ldots,N; i \neq k; j = 0,1,\ldots,L)$, usually caused by a change in the operative state (i.e. a change in random variable $I(t)$).

$B_j(i,k)$: One-step upward transition rate, from state (i,j) to state $(k,j+1)$, $(i = 0,1,\ldots,N; k = 0,1,\ldots,N;$ and $j = 0,1,\ldots,L)$, usually caused by a job arrival into the queue.

$C_j(i,k)$: One-step downward transition rate, from state (i,j) to state $(k,j-1)$, $(i = 0,1,\ldots,N; k = 0,1,\ldots,N;$ and $j = 0,1,\ldots,L)$, usually caused by the departure of a serviced job.

The Spectral Expansion method is applicable for systems with unbounded queuing capacities (i.e. $K \leq L < \infty$) as well as systems with bounded queuing capacities (i.e. finite $L \geq K$). The unbounded and bounded systems are presented in semi-finite and finite lattice strips respectively. The solution presented is also valid for steady states of multi-server systems.

Following the Spectral Expansion solution, the steady-state probabilities of the system considered can be expressed as:

$$P_{i,j} = \lim_{t \longrightarrow \infty} P(I(t) = i, J(t) = j); 0 \leq i \leq N, 0 \leq j \leq L$$

where L can be finite or infinite. Let's define certain diagonal matrices of size $(N+1)\times(N+1)$ as follows:

$$D_j^A(i,i) = \sum_{k=0}^{N} A_j(i,k);\ D^A(i,i) = \sum_{k=0}^{N} A(i,k)$$

$$D_j^B(i,i) = \sum_{k=0}^{N} B_j(i,k);\ D^B(i,i) = \sum_{k=0}^{N} B(i,k)$$

$$D_j^C(i,i) = \sum_{k=0}^{N} C_j(i,k);\ D^C(i,i) = \sum_{k=0}^{N} C(i,k)$$

and $Q_0 = B$, $Q_1 = A - D^A - D^B - D^C$, $Q_2 = C$. For both bounded and unbounded queuing systems, all state probabilities in a row can be defined as:

$v_j = (P_{0,j}, P_{1,j}, \ldots, P_{N,j}) j = 0,1,\ldots$

Here, for a bounded system, j is limited by finite L. In this case, when the queue is full, the arriving jobs are lost. The matrices given above are used in the Spectral Expansion solution for both bounded and unbounded queuing systems.

The steady-state balance equations for unbounded queuing systems can now be written as:

$$v_0[D_0^A + D_0^B] = v_0 A_0 + v_1 C_1 \tag{2.4}$$

$$v_j[D_j^A + D_j^B + D_j^C] = v_{j-1} B_{j-1} + v_j A_j + v_{j+1} C_{j+1}; 1 \leq j \leq M-1 \tag{2.5}$$

$$v_j[D^A + D^B + D^C] = v_{j-1} B + v_j A + v_{j+1} C; j \geq M \tag{2.6}$$

and the normalising equation is given as follows:

$$\sum_{j=0}^{\infty} v_j e = \sum_{j=0}^{\infty} \sum_{i=0}^{N} P_{i,j} = 1.0$$

from Equation (2.6) one can write

$$v_j Q_0 + v_{j+1} Q_1 + v_{j+2} Q_2 = 0; j \geq M-1$$

Furthermore, the characteristic matrix polynomial $Q(\lambda)$ can be defined as:

$$Q(\lambda) = Q_0 + Q_1 \lambda + Q_2 \lambda^2$$

$\lambda$and ψ are eigenvalues and left-eigenvectors of $Q(\lambda)$ respectively. Note that, ψ is a row-vector defined as

$\psi = \psi_0, \psi_1, \ldots, \psi_N$, $\lambda = \lambda_0, \lambda_1, \ldots, \lambda_N$ and $\psi Q(\lambda) = 0$; $|Q(\lambda)| = 0$.
Finally, for an unbounded system, when the stability condition is satisfied (Chakka 1995), one can obtain the general solution as:

$$v_j = \sum_{k=0}^{N} a_k \psi_k \lambda_k^{j-M+1}, j \geq M-1$$

and in the state probability form,

$$P_{i,j} = \sum_{k=0}^{N} a_k \psi_k(i) \lambda_k^{j-M+1}, j \geq M-1$$

where, $\lambda_k (k = 0, 1, \ldots, N)$ are $N+1$ eigenvalues that are strictly inside the unit circle (Chakka 1995) and $a_k (k = 0, 1, \ldots, N)$ are arbitrary constants which can be scalar or complex-conjugates. All the a_k values and the remaining v_j vectors can be obtained us-

ing the process in (Chakka 1995). For the case of bounded queue with $0 \leq j \leq L$, the balance equations are:

$$v_0[D_0^A + D_0^B] = v_0 A_0 + v_1 C_1 \tag{2.7}$$

$$v_j[D_j^A + D_j^B + D_j^C] = v_{j-1} B_{j-1} + v_j A_j + v_{j+1} C_{j+1}; 1 \leq j \leq M-1 \tag{2.8}$$

$$v_j[D^A + D^B + D^C] = v_{j-1} B + v_j A + v_{j+1} C; M \leq j < L \tag{2.9}$$

$$v_L[D^A + D^C] = v_{L-1} B + v_L A \tag{2.10}$$

The normalisation equation is given as:

$$\sum_{j=0}^{L} v_j e = \sum_{j=0}^{L} \sum_{i=0}^{N} P_{i,j} = 1.0$$

From Equation (2.9)

$$v_j Q_0 + v_{j+1} Q_1 + v_{j+2} Q_2 = 0; (M-1) \leq j \leq (L-2)$$

and the characteristic matrix polynomials can be expressed as:

$$Q(\lambda) = Q_0 + Q_1 \lambda + Q_2 \lambda^2; \overline{Q}(\beta) = Q_2 + Q_1 \beta + Q_0 \beta^2$$

where

$$\psi Q(\lambda) = 0; |Q(\lambda)| = 0; \phi \overline{Q}(\beta) = 0; |\overline{Q}(\beta)| = 0$$

β and ϕ are eigenvalues and left-eigenvectors of $\overline{Q}(\beta)$ respectively.Note that, ϕ is a vector defined as $\phi = \phi_0, \phi_1, \ldots, \phi_N$,$\beta = \beta_0, \beta_1, \ldots, \beta_N$.

Furthermore, $v_j = \sum_{k=0}^{N} (a_k \psi_k \lambda_k^{j-M+1} + b_k \phi_k \beta_k^{L-j})$, $M-1 \leq j \leq L$ and in the state probability form,

$$P_{i,j} = \sum_{k=0}^{N} (a_k \psi_k(i) \lambda_k^{j-M+1} + b_k \phi_k(i) \beta_k^{L-j}),\ M-1 \leq j \leq L$$

where $\lambda_k (k = 0, 1, \ldots, N)$ and $\beta_k (k = 0, 1, \ldots, N)$ are $N+1$ eigenvalues each, that are strictly inside the unit circle (Chakka 1995), and, $b_k (k = 0, 1, \ldots, N)$ are arbitrary constants which can be scalar or complex-conjugate,just like a_ks. v_js can be obtained as explained in the previous case.

From the $P_{i,j}$, a number of steady-state availability, reliability, performability measures can be computed quite easily. For example, mean queue length (MQL) can be obtained as:

$$MQL = \sum_{j=0}^{L} j \sum_{i=0}^{N} P_{i,j}$$

where L can be finite or infinite depending on whether the case concerned is bounded or unbounded. For cases where L is finite, the percentage of jobs lost (PJL) can be obtained by using the following equation:

$$PJL = 100 \times \sum_{i=0}^{N} P_{i,L}$$

Once the steady state probabilities are computed, it is possible to calculate some other system performance measures such as mean response time and throughput in addition to MQL and PJL shown above.

2.5 State Space Explosion Problem

Two dimensional models with multiple components are effectively used for the modelling of queuing systems with multiple queues and/or multiple servers. The functional equations arising in the analysis of such processes usually present significant analytic difficulties (Boxma et. al. 1994). These numerical difficulties are frequently caused by large number of system states. In other words, difficulty caused by the rapid increase in the size of the state space of the underlying Markov process is known as the state space explosion problem.

When two dimensional models (lattice strip) shown in figure 2.4 are considered, the cause of large number of state spaces can be either large (or infinite) number of $J(t)$ states or large number of $I(t)$ states.

For multi-dimensional models where one component is finite there are good analytic-algorithmic methods, like the ones given in previous sections (Matrix-geometric solution, Spectral Expansion method). These methods can be used to solve the state explosion problem caused

by large or infinite number of $J(t)$ states. In (El-Rayes et. al. 1999) to overcome the state explosion problem, a Stochastic Process Algebra based on Performance Evaluation Process Algebra (PEPA), called PEPA1ph, is formulated. PEPA1ph is used to specify systems with potentially, infinitely many customers in an infinite queueing system with durations given as phase-type distributions. Matrix-geometric solution method is used in order to calculate the steady state probability of the models that arise from a fragment of PEPA1ph.

Although systems with unbounded queuing capacities can be handled by the Spectral Expansion and the Matrix-geometric solution methods, as the number of $I(t)$ states increases they become computationally expensive. The numerical complexity of the solutions depends on the number of states $I(t)$ represents (Mitrani 2005). That number determines the size of the matrix R used in the Matrix-geometric method, and the number of eigenvalues and eigenvectors involved in the Spectral Expansion method. In both solution techniques the size of the matrices used depends on the number of states represented with $I(t)$. As the size of the matrices increase the computational requirements increase significantly. Because of the large size, ill-conditioned matrices numerical problems occur (Mitrani 2005). Also the numerical stability of the solution gets affected especially when heavily loaded systems are considered.

An approach is used in (Mitrani 2005) to overcome these problems. A simple geometric approximation is proposed to be used together with Spectral Expansion method to overcome computational difficulties.

The approach uses a *restricted* Spectral Expansion, based on a single dominant eigenvalue and its associated eigenvector. The eigenvalue provides a geometric approximation for the queue size distribution, while the eigenvector approximates the distribution of the $I(t)$. The dominant eigenvalue is the one nearest to but strictly less than 1. There are techniques for determining the eigenvalues that are near a given number (Mitrani 2005). The approach avoids completely the need to solve a set of linear equations which also means avoiding all problems associated with ill-conditioned matrices. Multi-server systems with breakdowns and repairs (Figure 2.1) and Tandem systems (Figure 2.7) are used in order to evaluate the accuracy of the new approach. Poisson arrivals and exponentially distributed service times are considered for both systems. The limiting effect of state explosion problem is greater for Tandem systems since $I(t)$ is used to represent the number of jobs in the queue. In (Mitrani 2005) it is stated that the Spectral Expansion method begins to experience numerical difficulties when the queuing capacity extends 35, and the software (Matlab) starts issuing warnings to the effect that the matrix is ill-conditioned, and the results may not be reliable. The given results show that this approach works well for the systems under

heavy traffic. However when relatively lower arrival rates are considered the accuracy of approximations decreases.

It is important to solve the numerical difficulties caused by large number of $I(t)$ states especially for Tandem systems. Because of the limitations, it is difficult to consider more than one server at stage 2 (Figure 2.7) which can be used for representation of many practical systems (e.g. communication channels and multi-server systems in tandem). This is because the possible inclusion of multiple servers, breakdowns, and repairs will increase the complexity of $I(t)$ and will severely limit the queue capacity. This is an important research item which is addressed in this thesis.

2.6 Conclusion

In the literature review section various evaluation methods for multi-server systems are described and analysed. Basic terms and techniques used in this research are defined. Pure performance, pure availability/reliability and performability modelling techniques are studied. The existing modelling methods developed for each kind of evaluation techniques are critically analysed and compared. As a result, it has been seen that composite measure of performance and availability is the most realistic, accurate modelling technique for multi-server systems, since these systems are usually fault-tolerant.

Background information of systems under study is given. Existing models for performability evaluation of these systems are critically analysed. The main gaps of existing models are underlined. The necessity of having multi-dimensional representations for handling the irregularities caused by failure/repair phenomena is stated. The existing methods to solve multi-dimensional state spaces, for their steady state probabilities are critically compared. The advantages of using Spectral Expansion method are underlined. Also a brief explanation of Spectral Expansion method is given.

The well known state space explosion problem is underlined and existing solution methods are investigated in the last section.

Chapter 3

The Effects of Reconfiguration and Rebooting Delays In Performability Evaluation of Multi-Server Systems with Breakdowns

In practice when fault-tolerant multi-server systems are considered, some delays are encountered when a failed server is being mapped out of the system or a repaired server is being admitted into the system.

Since these systems are prone to breakdowns these delays can have significant effects on systems performability depending on the nature of system. Furthermore, these delays can be categorised as reconfiguration and rebooting delays depending on the cover facilities of the system (Trivedi et. al. 1990). Subsequent to a covered fault, the system comes up in a degraded mode after a brief delay. These delays are called reconfiguration delays. On the other hand if the fault is uncovered a longer action is required to bring the system up at a degraded mode. These delays are significantly larger than reconfiguration delays and they are called rebooting delays. In this study the parameter c is used to represent the probability of having reconfiguration delays (fault covered). Then $1 - c$ can be used to represent the probability of having rebooting delays (fault is uncovered). Here, degraded mode indicates a state with one less operative server than the previous state.

3.1 Homogeneous Multi-server Systems with Reconfiguration and Rebooting Delays

In this section approaches are developed to model homogeneous multi-server systems with reconfiguration and rebooting delays by suitably extending the resulting QBD process in the performance models of multiprocessor systems with breakdowns and repair strategies (Chakka and Mitrani 1992, Chakka et. al. 2002).

A similar study is considered in (Trivedi et. al. 1990). An approximate performance model based on Markov reward models was presented. In this section, an exact solution for the steady state probabilities of the same problem using the Spectral Expansion method is derived. The effects of reconfiguration and rebooting delays are analysed together with other queuing issues such as various failure rates, finite and infinite queuing capacities.

3.1.1 Multi-Server System with Identical Processors and Reconfiguration/Rebooting Delays

The performance modelling of a multiprocessor system, with identical servers, serving a stream of arriving jobs is considered. The system considered is shown in figure 3.1.

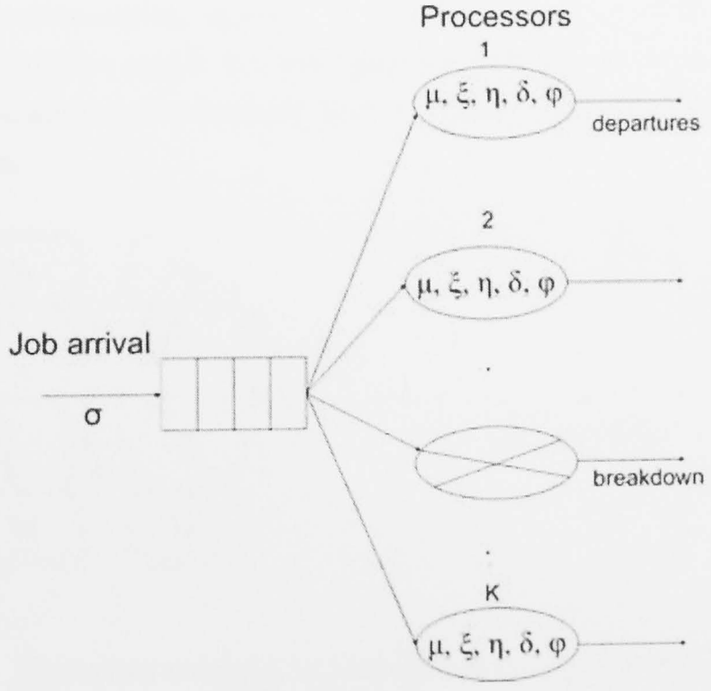

Figure 3.1: A homogeneous multi-server system with breakdowns, repairs, reconfiguration and rebooting delays

The homogeneous multi-server system has K servers. Jobs arrive at the system in a Poisson stream and join the queue (Hacker and Athey 2001, Trivedi 2002, Gyu et. al. 2004). The mean arrival rate of jobs is given by σ. The queuing capacity is L, where L can be finite or infinite.

Parameters μ and ξ are the mean service and mean failure rates of each of the servers. There is a single repair facility and the repair rate is given by η. In case of server failures the fault is covered with probability c and the system comes up in a degraded mode after a brief reconfiguration delay. On the other hand if the fault is not covered with probability $1 - c$ a longer reboot action takes place in order to bring the system up at a degraded mode. For reconfiguration and rebooting periods, the system is assumed to be down (Trivedi et. al. 1990). The reconfiguration and rebooting times are exponentially distributed as well. The mean values of reconfiguration and rebooting times can be given as $1/\delta$ and $1/\varphi$ respectively. All inter-arrival, service, reconfiguration, rebooting, operative and repair time random variables are independent of each other.

3.1.2 Modelling Multi-Server Systems with Identical Servers

A Markov chain analysis is carried out in (Trivedi et. al. 1990) for computing the dependability of a multiprocessor system. It is possible to use a similar Markov chain in order to present the operative states of a homogeneous multi-server system with breakdowns, repairs, and reconfiguration and rebooting delays. Figure 3.2 shows the possible operative states of such a system.

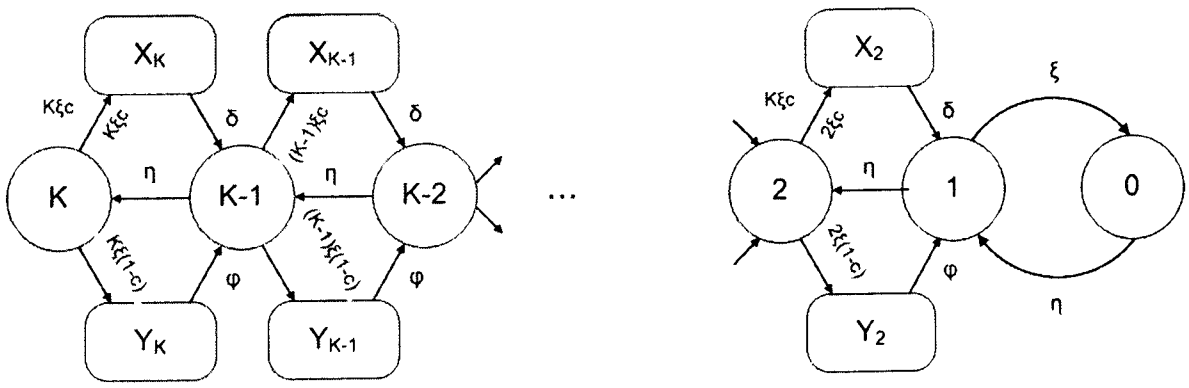

Figure 3.2: Operative states of a homogeneous multiprocessor system with breakdowns, repairs, reconfiguration and rebooting delays

The total number of operative states is $3K-1$. In state 0 there are no operational servers. The states labelled as 1, 2, ..., K are the K states where the servers are operative, with that many number of servers in each state. The $K-1$ states, labelled as X_2, X_3, ..., X_K are the states which represent the reconfiguration delays. The last $K-1$ states labelled as Y_2, Y_3, ..., Y_K are the rebooting delay states.

It is possible to use a two dimensional representation similar to one explained in section 2.2.3 for this system. With this approach, the system can be represented by a QBD process with finite or infinite state space. The state of the system can be defined by $(I(t), J(t))$ where $I(t)$ is the operative state and $J(t)$ is the number of jobs in the system.

In order to use the Spectral Expansion method A, B and C matrices should be defined. For this system the elements of A depend on the parameters K, ξ, η, c, δ and φ. Where the elements of B depend on arrivals, and elements of C depend on service times of the servers. Numbering of the operative states can be arbitrary. First K numbers (0, 1, ..., K) are given to states 0, 1, 2, ..., K. Following $K-1$ numbers ($K+1$, $K+2$, ..., $2K-1$) are given to states X_2, X_3, ..., X_K. The last $K-1$ numbers ($2K$, $2K+1$, ..., $3K-2$) are given to states Y_2, Y_3, ..., Y_K. Then, state transition matrices A, A_j, B, B_j, C, and C_j, can be given as follows;

$$A_j = A = \begin{pmatrix}
0 & \eta & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\
\xi & 0 & \eta & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \eta & 0 & 0 & \cdots & 2\xi c & 0 & 0 & 0 & \cdots & 2\xi(1-c) & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \ddots & 0 & \cdots & 0 & 3\xi c & 0 & 0 & \cdots & 0 & 3\xi(1-c) & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \eta & 0 & \cdots & 0 & \ddots & 0 & \cdots & 0 & 0 & \ddots & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K\xi c & \cdots & 0 & 0 & 0 & K\xi(1-c) \\
0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \ddots & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \ddots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

$B_j = B$, for $j = 0, 1, \ldots, L$ and B is a diagonal matrix defined as $B = Diag[\sigma, \sigma, \ldots, \sigma]$ of size $(3K-1) \times (3K-1)$.

Since the parameters of A and B matrices are independent from the number of jobs in the system, $A_j = A$ and $B_j = B$. On the other hand the service rate provided depends on the number of jobs in the system if the number of jobs in the system is less than the threshold value M, where $M = K$. Then, the matrices C and C_j can be given as:

$C_j = C$ for $j \geq K$ and C is a diagonal matrix defined as
$C = Diag[w(0)\mu, w(1)\mu, \ldots, w(3K-2)\mu]$ of size$(3K-1) \times (3K-1)$.
$C_j = Diag[Min(w(0), j)\mu, Min(w(1), j)\mu, \ldots, Min(w(3K-2), j)\mu]$ for $1 \geq j \geq K$ and
$C_0 = [0]$ where $w(i)$ is the number of working processors in the operative state i. The last $2K-2$ elements of C and C_j matrices are zero since they are used for reconfiguration and rebooting delay states.

3.1.3 Numerical Results for Multi-Server Systems with Identical Servers

In order to show the the effectiveness of the method presented and to analyse the effects of reconfiguration, and rebooting delays on performability of homogeneous multi-server systems, numerical results are presented in this section. The other queuing system characteristics such as various queue sizes, systems with various number of servers are also taken into account.

First the numerical results are presented for systems with unbounded queues. Systems with various number of servers are considered below. Parameters are given as σ jobs/sec, $\xi = 0.001$/hr, $\eta = 0.2$/hr, $\mu = 4000$ jobs/hr, $\varphi = 2$/hr, and $\delta = 60$/hr, $c = 0$ (only rebooting).

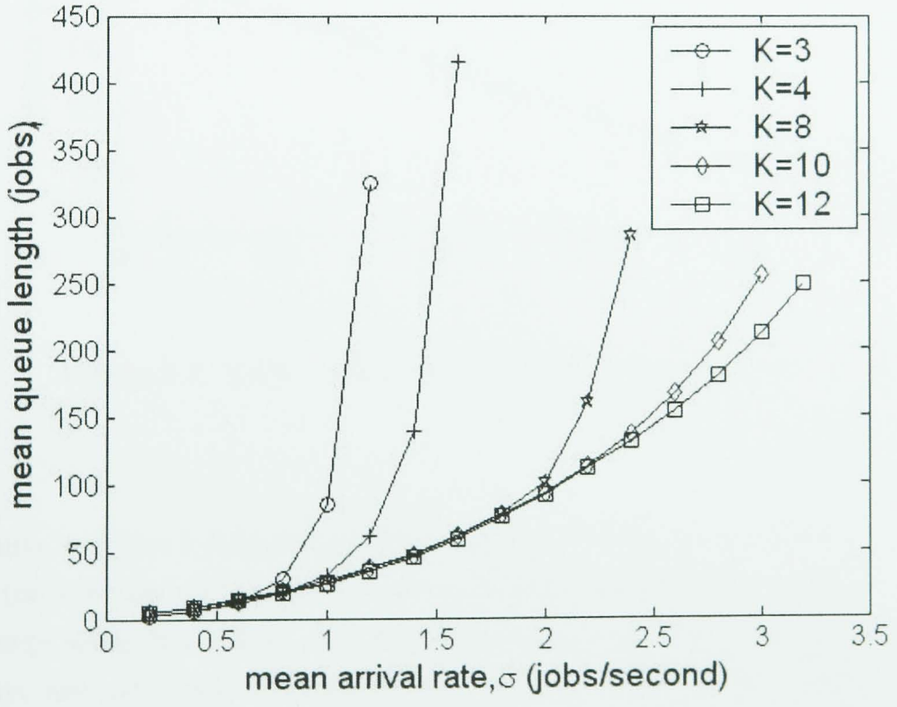

Figure 3.3: MQL versus σ for homogeneous systems with various K

The results in figure 3.3 show that when systems with infinite queuing capacities are considered, systems with larger number of servers perform significantly better especially when the system is heavily loaded (high arrival rates are expected).

In order to show the effects of c on systems with various number of servers, the mean queue length is calculated as a function of c where $\sigma = 20$ jobs/sec, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\varphi = 2$/hr, and $\delta = 60$/hr. The service rates are taken as $\mu = \sigma/(0.7K)$ in order to provide constant utilisation (in case of no breakdowns) which is useful to analyse the cost effectiveness of the systems. The results are illustrated in figure 3.4.

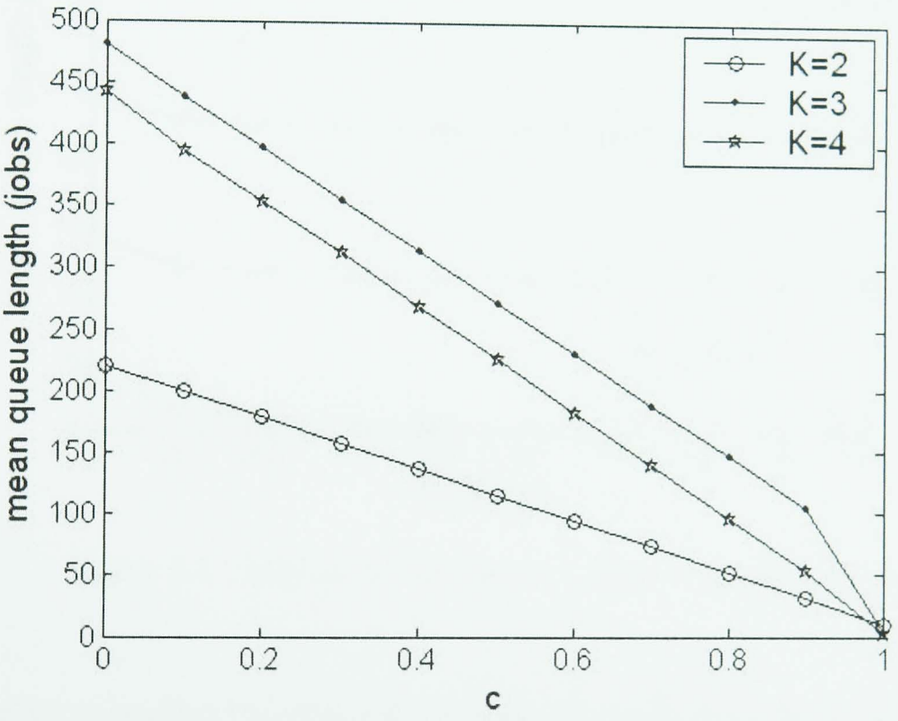

Figure 3.4: MQL versus c for homogeneous systems with various K

The results in figure 3.4 show that when constant utilisation is considered (assume no breakdowns) the systems with smaller number of servers can perform better. This is mainly due to the large rebooting delays. The figure clearly shows that when c is less than one (rebooting delays are expected) the two server system performs better than both four and three server systems. This is because in a two server system there is only one reconfiguration and rebooting state. Four server system performs better than three server system but worse

that two server system when c is less than one. On the other hand when $c = 1$, since only brief reconfiguration delays are expected, the systems with larger number of servers perform better.

The mean queue length is computed as a function of reconfiguration rate for δ for a four server system in figure 3.5. Other parameters are $\sigma = 1$ job/sec, $\mu = \sigma/(0.7K)$, $\xi = 0.001$/hr, $\eta = 0.5$/hr and $\varphi = 2$/hr.

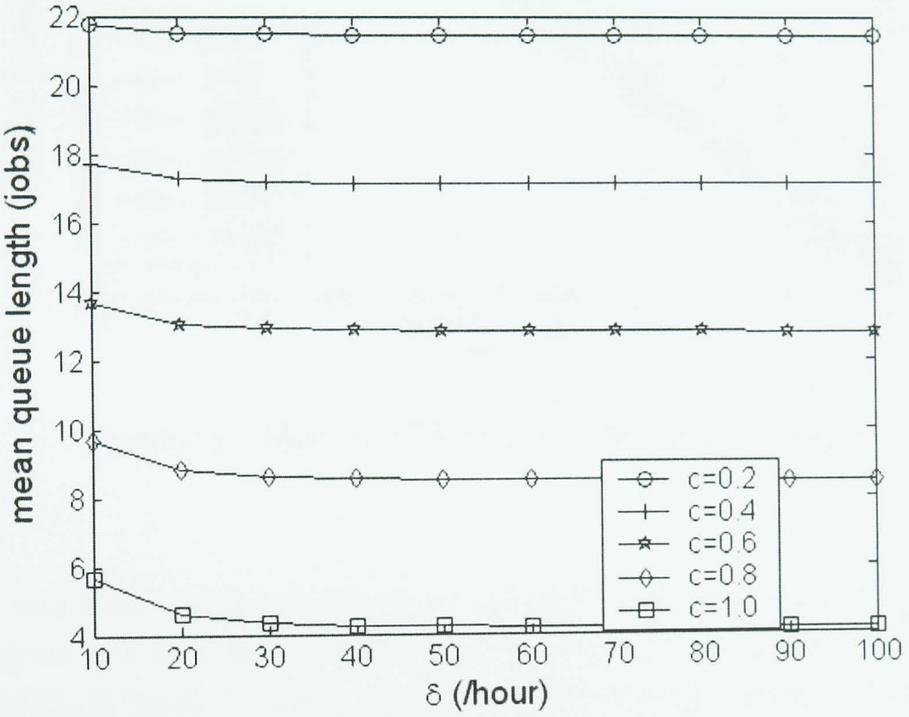

Figure 3.5: MQL as a Function of δ for homogeneous systems

The figure 3.5 shows that the effects of δ becomes more significant as c grows. Also from this figure it is evident that after a certain point ($\delta = 60$ for this computation) decreasing the reconfiguration delay (increasing δ) does not affect the performance of the system. Performance measures of the system approaches to the performance measures of a system without any delays.

In figure 3.6 a system with four servers is considered. MQL is computed as a function of c for various δ values where $\sigma = 1$ job/sec, $\mu = \sigma/(0.7K)$, $\xi = 0.001$/hr, $\eta = 0.5$/hr and $\varphi = 2$/hr. The results show that for larger δ the effects of c is more significant. Also similar to the previous figure the systems performance does not change after a certain δ value ($\delta = 20$ in this figure).

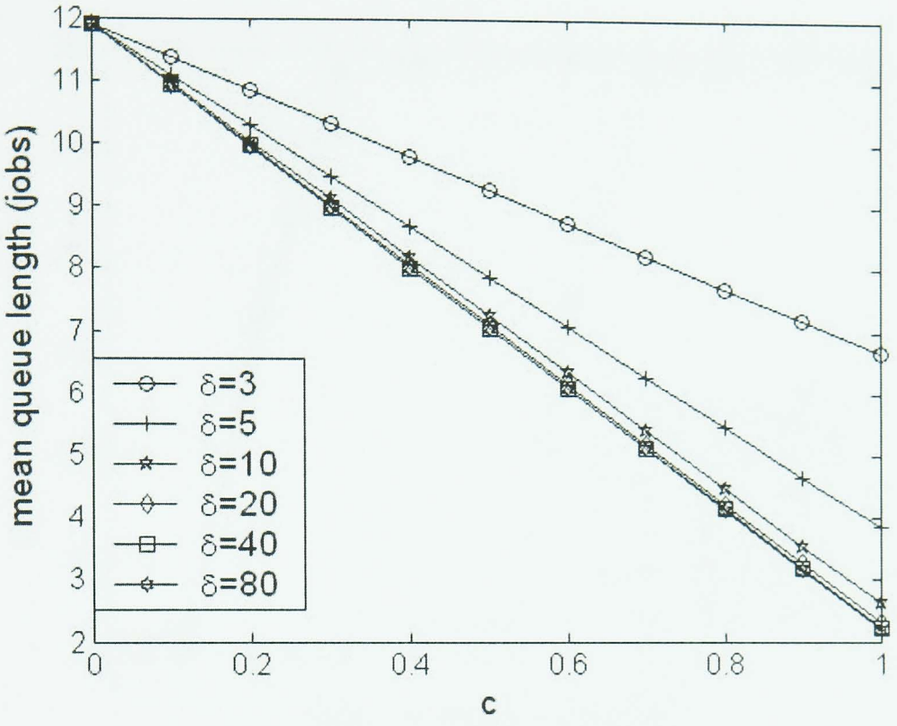

Figure 3.6: MQL as a Function of c for homogeneous systems

Systems with finite queuing capacities are considered for the following figures. The parameters are given as σ jobs/hr, $\xi = 0.001$/hr, $\eta = 0.2$/hr, $\mu = 2$ jobs/hr, $\varphi = 2$ /hr, $\delta = 60$ /hr, and $L = 300$. In figure 3.7 MQL values and in figure 3.8 percentage of jobs lost (PJL) are presented.

Figures 3.7and 3.8 show that the effect of c on systems performance is not significant for systems with bounded queues. This means that when systems with finite queuing capacities are considered, the cover facilities do not affect the systems performability due to the limiting effect of queue size. Also, when heavily loaded systems are considered, having an extra server does not necessarily improve the performability in terms of MQL. For relatively low arrival rates, systems with four servers have better MQL values than systems with three servers, but when high arrivals are expected ($\sigma > 8.5$ in this case) the queue capacity is the main limiting factor. On the other hand when PJL values are considered, four server systems have lower loss rates.

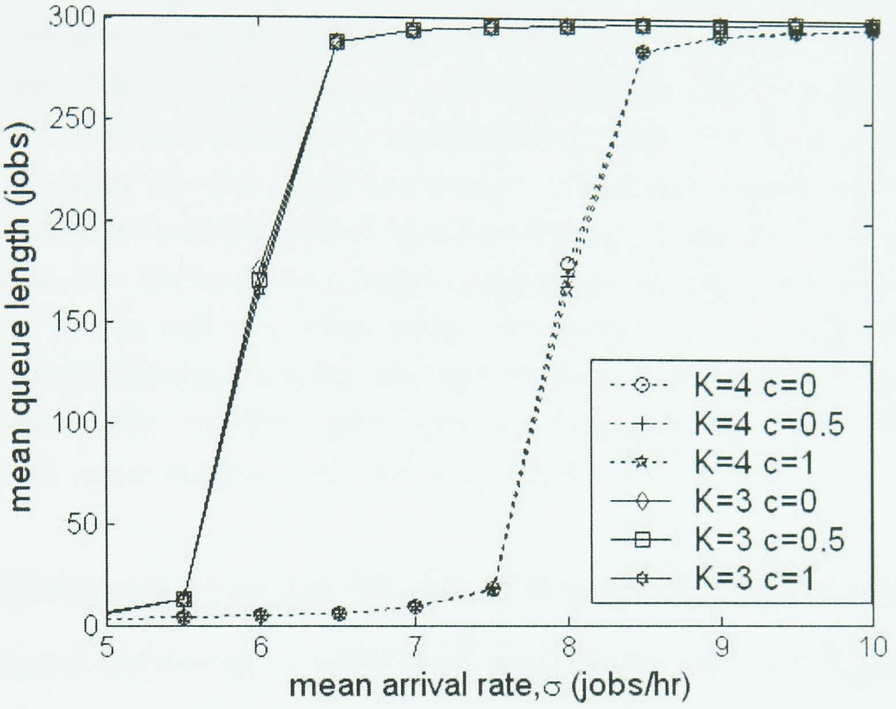

Figure 3.7: MQL as a Function of K, c, and σ for homogeneous systems with L = 300

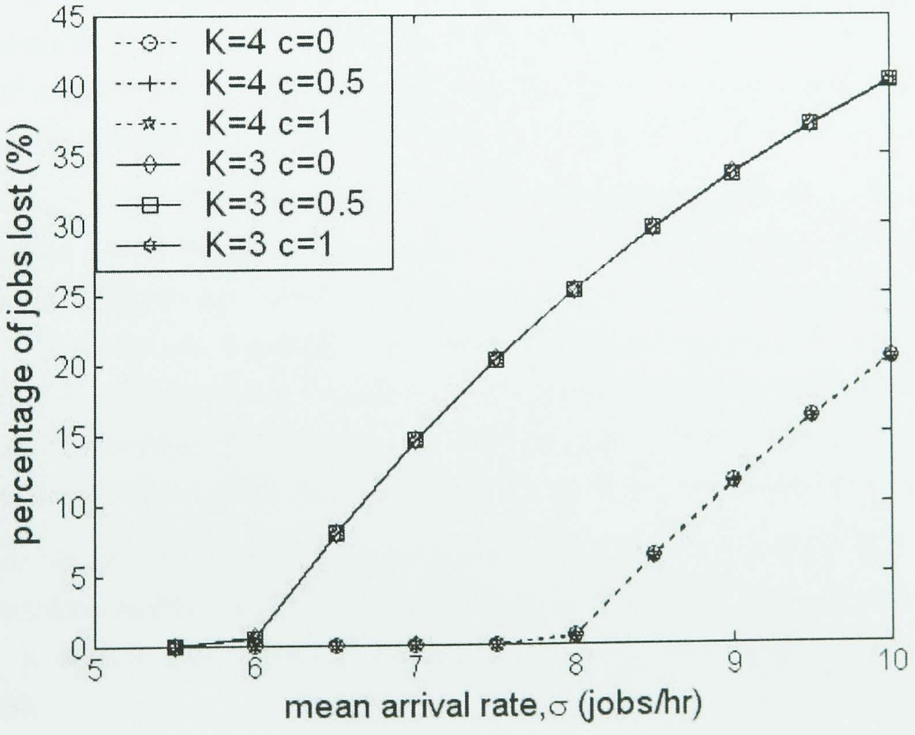

Figure 3.8: PJL as a Function of K, c, and σ for homogeneous systems with L = 300

In this section homogeneous multi-server systems with breakdowns and repairs are modeled for exact solution. Since the systems considered are fault-tolerant, considering reconfiguration and rebooting delays will make the performability analysis of such systems more realistic. The model is providing flexibility to analyse such systems with various characteristics, while the queuing issues are also taken into account. Numerical results obtained show that the reconfiguration and rebooting delays may have significant effects on system's performability. Furthermore, the choice of the optimum number of processors may depend on the values of $1/\delta$, and $1/\varphi$ as well as c, when service rate is taken as a function of arrival rate. For bounded queuing systems, L is the main factor affecting the mean queue length performance of the system when relatively high arrival rates are expected. When PJLs are compared, systems with larger number of servers perform better.

3.2 Heterogeneous Multi-Server Systems with One Main and Several Identical Servers, Reconfiguration and Rebooting Delays

Practically servers used in a single system may be non-identical. One example can be given as a system with a main server and several identical servers. There are examples of such systems especially in farm processor systems (Wagner et. al. 1997, Adams and Vos 2002). In case of main server breakdowns, the identical servers keep providing service. The main server usually has greater service power than the identical ones (Adams and Vos 2002).

It is possible to extend the approach given for homogeneous systems, to model systems with one main and several identical servers for performability evaluation. The difference between homogeneous systems and these systems is the main server with possibly different service rate. In this section an approach is developed for performability analysis of a specific type of heterogeneous multi-server system with reconfiguration and rebooting delays by suitably extending the resulting QBD process in the performance models of multi-server systems with breakdowns and repair strategies (Chakka and Mitrani 1992, Chakka et. al. 2002).

This time the multi-server system considered, consists of one main and $K-1$ identical parallel servers, numbered $1, 2, \ldots, K$ with a common queue. The queue can have finite (L, $L \geq K$) or infinite size. Model is applicable for both cases. Figure 3.9 shows the system considered.

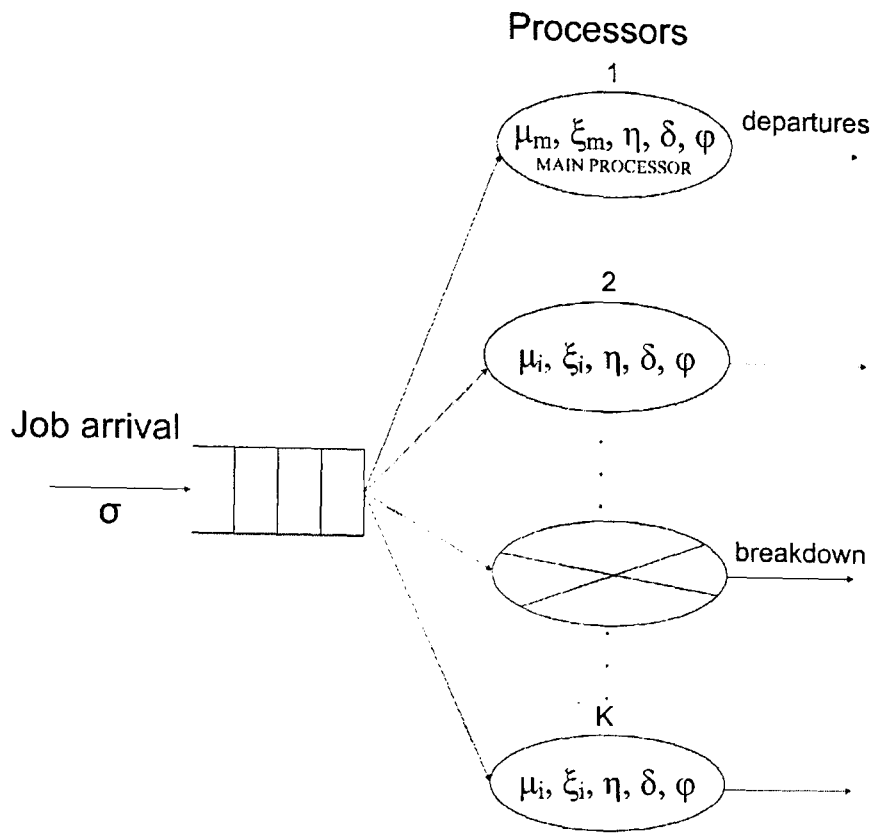

Figure 3.9: A heterogeneous multi-server system with one main and $K-1$ identical servers

Similar to the homogeneous systems, the jobs arrive at the system in a Poisson stream at mean rate σ, and join the queue. Jobs are homogeneous and the identical servers have the same service rates. The service times are distributed exponentially and mean service rate of the main server is usually greater then the identical ones. The service times of jobs serviced by identical server k (i.e. $k = 2, \ldots, K$) are distributed with mean $1/\mu_i$ and service times of jobs serviced by the main server (i.e. $k = 1$) are distributed with mean $1/\mu_m$. The operative periods of main and identical servers are distributed with means $1/\xi_m$ and $1/\xi_i$ respectively. At the end of an operative period, server k ($k = 1, 2, \ldots, K$) breaks down and requires repair. The repair time is distributed with mean $1/\eta$. The repair times of main and identical servers are the same since the same facility is used for both. There is a single repair facility which can work on at most, one repair at a time. This means that, an inoperative period of a server also includes the possible waiting time for the repairman. If the main server is broken repair priority is given to this server. If the main server is not broken and there are more than one identical servers which are broken, the first server to repair, is chosen randomly. No operative server can be idle if there are jobs awaiting service, and the repairman cannot be idle if there are broken-down servers waiting for repair. The reconfiguration and rebooting

delays are distributed with means $1/\delta$ and $1/\varphi$ respectively. These delays are assumed same for all servers since they are related to the system and not to individual servers. However, it is to be noted that the methodology developed here is applicable even if these delays are different for the main and the identical servers.

3.2.1 Modelling Multi-Server Systems with One Main and Several Identical Servers

In order to represent such a system by a QBD process with finite or infinite state space, the operative states of the system should be specified. The state of the system is defined by $(I(t), J(t))$ where $I(t)$ represents the operative states of the system and $J(t)$ is the number of jobs in the system. In this section, a Markov chain is given for the operative states of heterogeneous multi-server system considered.

The reconfiguration and rebooting delays are also taken into account in order to make the performability evaluation approach more realistic. It is possible to model the system affected by such reconfiguration and rebooting delays effectively, by using the Spectral Expansion method. Here parameter c is again used to represent the probability of having reconfiguration delays and $1-c$ is used to represent the probability of having rebooting delays when a server breaks down. The failures may occur at the main server or at one of the identical servers. The main server can work in the absence of identical servers and identical servers are capable of serving when the main server is broken. For a reconfiguration/rebooting period, the system is completely down (Trivedi et. al. 1990).

The performance modelling presented in the previous section can be extended in order to have a valid model for systems with one main and several identical servers. Figure 3.10 is the Markov chain that represents the operative states $(I(t))$ of the heterogeneous multi-server system discussed.

In state 0 there are no operational servers. The states labelled as 1_m, 2_m,..., K_m are the states of the multi-server system where the main server is operational. The number shows the total number of operational servers including the main server (i.e. K_m means 1 main and $K-1$ identical servers). The states labelled 1_i, 2_i,..., $(K-1)_i$ are the states, where the main server is not operative and the number indicates the number of operational identical servers. There are no repair transitions between states which main server is not operative, since the repair priority is given to the main server. The first repair transition always brings the main server back. Also the breakdown transitions which makes the system totally non-operative (to state 0) do not require reconfiguration or rebooting delay states, since the system will not come up in a degraded mode.

The states, labelled as X_{1m}, X_{2m}, ..., $X_{(K-1)_m}$, and Y_{1m}, Y_{2m}, ..., $Y_{(K-1)_m}$ are the states representing the reconfiguration and rebooting delays respectively which is subsequent to an identical server breakdown while the main server is operative. The states, labelled as X_{1i}, X_{2i}, ..., $X_{(K-1)_i}$, and Y_{1i}, Y_{2i}, ..., $Y_{(K-1)_i}$ are the states representing the reconfiguration and rebooting delays respectively which is subsequent to an identical server breakdown while the main server is not operative. Finally, states X_{1t}, X_{2t}, ..., $X_{(K-1)_t}$, and Y_{1t}, Y_{2t}, ..., $Y_{(K-1)_t}$ represent the reconfiguration and rebooting delays which occur when the main server fails.

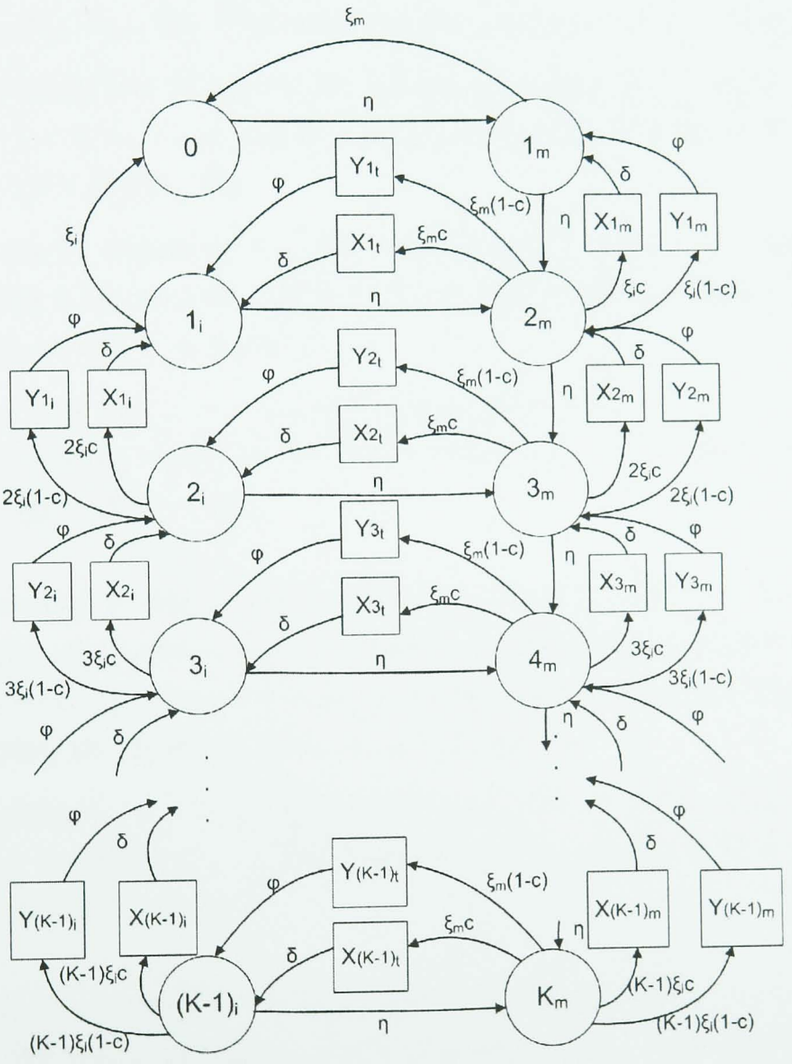

Figure 3.10: Operative states of a heterogeneous multi-server system with one main and $K-1$ identical servers

For this system the elements of matrix A include the failure rates of main and identical servers, repair rate, reconfiguration/rebooting delays, c and K which is used to specify the

size of the matrices. The elements of matrix B include mean arrival rates, and the elements of matrix C include mean service rates of main and identical servers. The total number of states is given as $8(K-1)$. For the numbering of the operative states first $2K$ numbers (0, 1, ..., $2K-1$) are given for states 0 and states with one or more operational servers. Odd numbers are given to states with operative main server and even numbers are given for states where the main server is not operative. The remaining part is used for reconfiguration and rebooting delays respectively. For example for a four server system the order of the states is taken as (0, 1_m, 1_i, 2_m, 2_i, 3_m, 3_i, 4_m, X_{1m}, X_{1i}, X_{1t}, X_{2m}, X_{2i}, X_{2t}, X_{3m}, X_{3t}, Y_{1m}, Y_{1i}, Y_{1t}, Y_{2m}, Y_{2i}, Y_{2t}, Y_{3m}, Y_{3t}. The state transition matrices B, B_j, C, and C_j, are given below:

The parameters of the B matrix are independent from the number of jobs in the system. $B_j = B$, for $j = 0, 1, \ldots, L$ and B is a diagonal matrix defined as $B = Diag[\sigma, \sigma, \ldots, \sigma]$ of size $(8(K-1)) \times (8(K-1))$.

For the matrix C, similar to the previous section the service rate provided depends on the number of jobs if the number of jobs in the system is less than the threshold value M, where $M = K$. The matrix C is given as :

$C_j = C$ for $j \geq K$ and C is a diagonal matrix defined as
$C = Diag[(w_m(0)\mu_m + w_i(0)\mu_i), (w_m(1)\mu_m + w_i(1)\mu_i), \ldots, (w_m(8K-9)\mu_m + w_i(8K-9)\mu_i)]$of size $(8(K-1)) \times (8(K-1))$.

When the number of jobs in the system is less than the total number of servers the priority is given to the main server since it usually provides higher service power. The first job goes to the main server (if operative) and the remaining ones go to the identical ones. If the main server is broken the server to use is chosen randomly.

Then C_j becomes $C_j = Diag[(Min(w_m(0), j)\mu_m + Min(w_i(0), j - w_m(0))\mu_i), (Min(w_m(1), j)\mu_m + Min(w_i(1), j - w_m(1))\mu_i), \ldots, (Min(w_m(8K-9), j)\mu_m + Min(w_i(8K-9), j - w_m(8K-9))\mu_i)$ for $1 \geq j \geq K$ and $C_0 = [0]$.

where $w_m(i)$ is the number of main servers and $w_i(i)$ is the number of identical servers working in the operative state i. Since there is only one main server in the system $w_m(i)$ is either 0 or 1 (main server broken or operative). The last $6K-8$ elements of C and C_j matrices are zero since they are used for reconfiguration and rebooting delay states.

The matrix A is also independent from the number of jobs in the system. This matrix is too bulky to be written in compact form, as an example, for $K = 4$, A and A_j can be written as follows:

$$A_j = A =$$

$$\begin{pmatrix}
0 & \eta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_m & 0 & 0 & \eta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_i & 0 & 0 & \eta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \eta & 0 & 0 & \xi_i c & 0 & \xi_m c & 0 & 0 & 0 & 0 & 0 & \xi_i(1-c) & 0 & \xi_m(1-c) & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \eta & 0 & 0 & 0 & 2\xi_i c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\xi_i(1-c) & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \eta & 0 & 0 & 0 & 2\xi_i c & 0 & \xi_m c & 0 & 0 & 0 & 0 & 0 & 2\xi_i(1-c) & 0 & \xi_m(1-c) & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \eta & 0 & 0 & 0 & 0 & 3\xi_i c & 0 & 0 & 0 & 0 & 0 & 0 & 3\xi_i(1-c) & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3\xi_i c & \xi_m c & 0 & 0 & 0 & 0 & 0 & 0 & 3\xi_i(1-c) & \xi_m(1- \\
0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

This system is solved by using the Spectral Expansion method. The steady state probabilities, $P_{i,j}$, are used for performability measures.

3.2.2 Numerical Results for Heterogeneous Systems with one Main and Several Identical Servers

Numerical results are presented in this section to show the effectiveness of the model developed for heterogeneous multi-server systems with one main and several identical processors, and to evaluate the performability of these systems. The systems with unbounded queuing capacities are considered first. In Figure 3.11, 2, 3, and 4-server systems are considered where parameters are given as σ jobs/sec, $\xi_m = \xi_i = 0.001$/hr, $\eta = 0.5$/hr, $\mu_i = 4000$ jobs/hr, $\mu_m = 8000$ jobs/hr, $c = 0$, $\varphi = 2$/hr, and $\delta = 60$/hr.

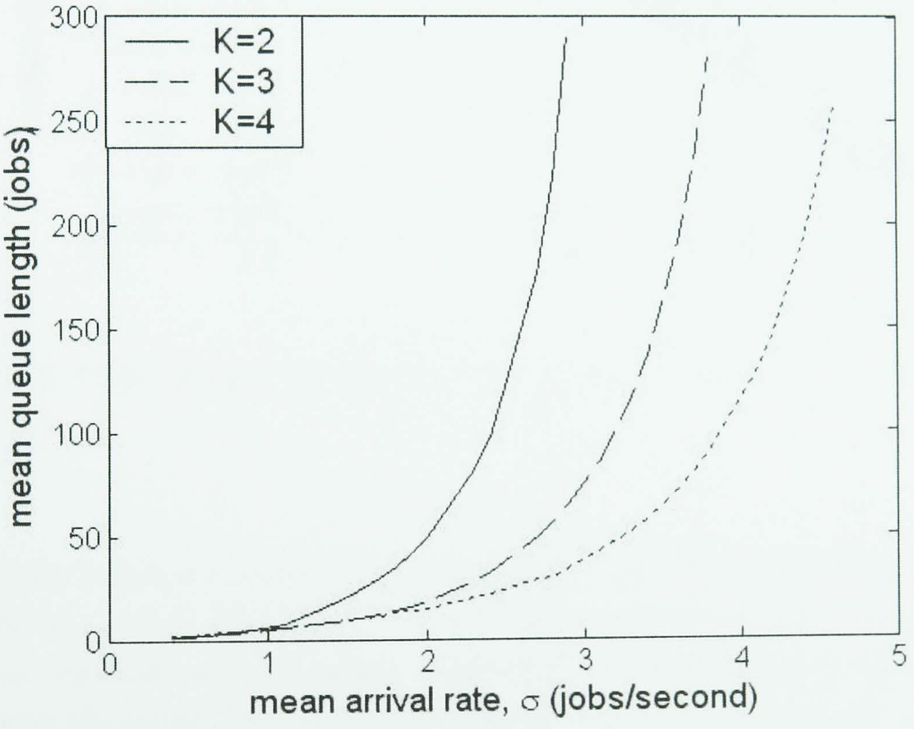

Figure 3.11: MQL versus σ for heterogeneous systems with one main and several identical servers

The results clearly show that when relatively low arrival rates are considered, systems have close MQL performances. This is because the server utilisations are low, and hence the queuing time of a job is short. When the arrival rate increases the systems with larger numbers of servers perform better.

A three server system is considered in figure 3.12. Other parameters are $K = 3$, $\sigma = 2$ jobs/sec, $\xi_m = \xi_i = 0.001$/hr, $\eta = 0.5$/hr, $\mu_i = 4000$ jobs/hr, $\mu_m = 8000$ jobs/hr, $\varphi = 2$ /hr, and $\delta = 60$ /hr. Effects of various reconfiguration delays are investigated.

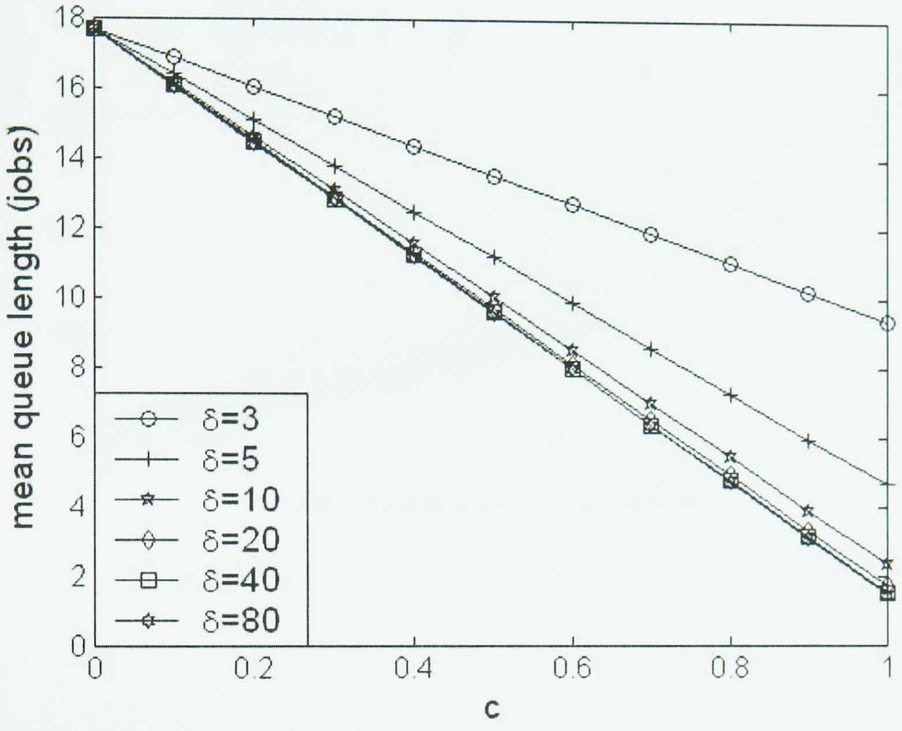

Figure 3.12: MQL versus c for heterogeneous systems with one main and several identical servers

The system's behavior is similar to homogeneous system when MQL values are computed as a function of c and δ. The effects of c increases as δ value increases. After a certain value ($\delta = 10$ in this case) various δ values do not affect the performability significantly. Also the figure shows the importance of the cover facility.

The effects of having various service rates for the main server is analysed in the following figure where $K = 3$, σ jobs/sec, $\xi_m = \xi_i = 0.001$/hr, $\eta = 0.5$/hr, $\mu_i = 4000$ jobs/hr, $c = 0$, $\varphi = 2$ /hr, and $\delta = 60$ /hr. The service rate of the main server is taken greater than the service rate of identical ones.

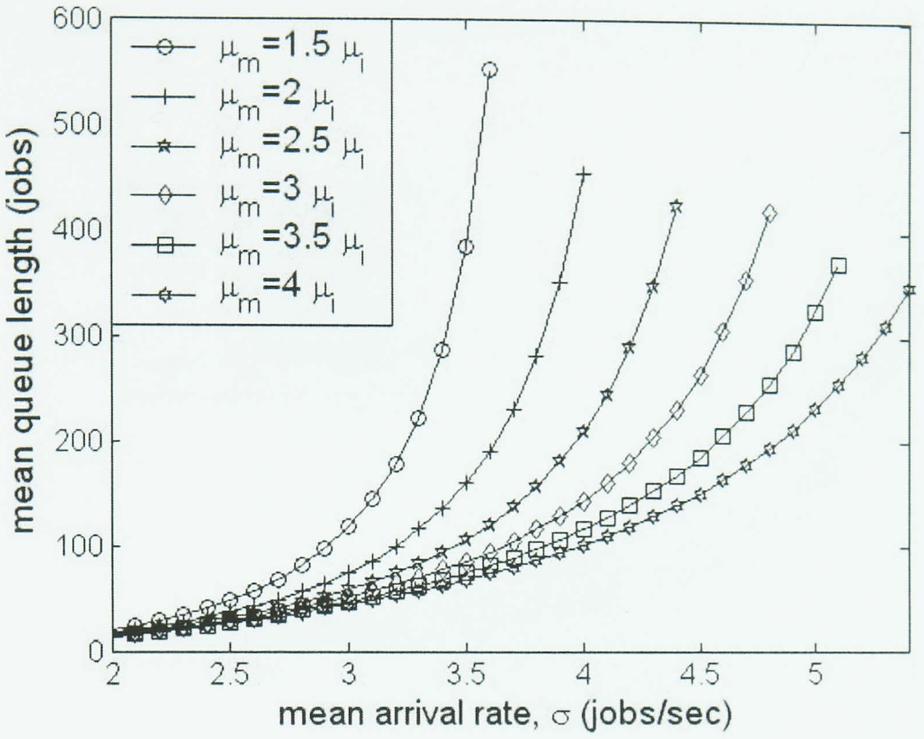

Figure 3.13: MQL versus σ for various μ_m values

In figure 3.13, since the system considered is a small scale system having a main server with larger service rates affects the systems performance significantly. When light loaded systems are considered the performance difference is not significant. However, when heavily loaded systems are considered the difference is more significant.

The model gives the flexibility to have different failure rates for the main and identical servers. In order to analyse the effects of various failure rates figure 3.14 is given. Other parameter are $K = 3$, σ jobs/sec, $\xi_i = 0.001$/hr, $\eta = 0.5$/hr, $\mu_i = 4000$ jobs/hr, $\mu_m = 8000$ jobs/hr, $c = 0$, $\varphi = 2$ /hr, and $\delta = 60$ /hr.

The effect on the reliability of the main server is illustrated in figure 3.14. In the small scale multi-server system considered, having various failure rates for the main server, which has greater service rates affects the systems performance significantly.

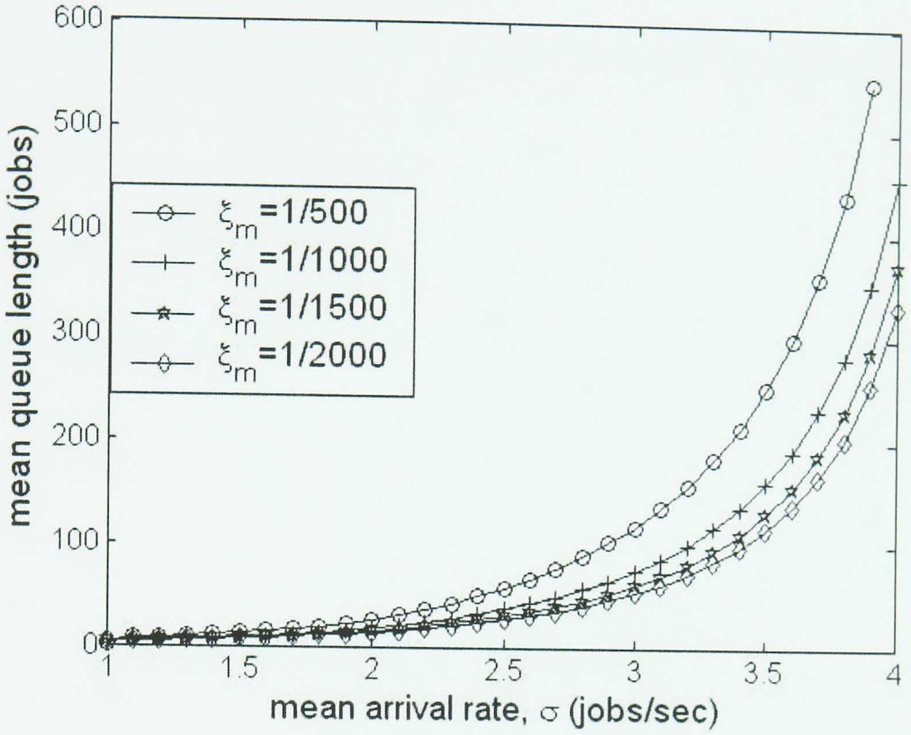

Figure 3.14: MQL versus σ for various ξ_m values

The following results are presented to demonstrate the effects of finite queuing capacity on heterogeneous multiprocessor systems with one main, and several identical processors. Also effects of having a main processor which provides higher service power are analysed together with bounding effect of limited queuing capacity.

Three server systems are considered in figure 3.15 where parameters are given as σ jobs/hr, $\xi_m = \xi_i = 0.001$/hr, $\eta = 0.5$/hr, $\mu_i = 10$ jobs/hr, $c = 0$, $\varphi = 2$ /hr, and $\delta = 60$ /hr. The results show that when highly loaded systems are considered, and the MQL values reach the queue size limit, having various service rates for the main server does not affect the systems MQL performance. However, when systems with relatively lower arrival rates are considered, having higher service rates for the main server improves the MQL performance significantly since the system considered is a small scale system ($K = 3$). Also the points that MQL reaches queue size varies depending on the service power of the main processor. Same parameters are used to compute the PJL as a function of σ. Figure 3.16 illustrates the results of this computation.

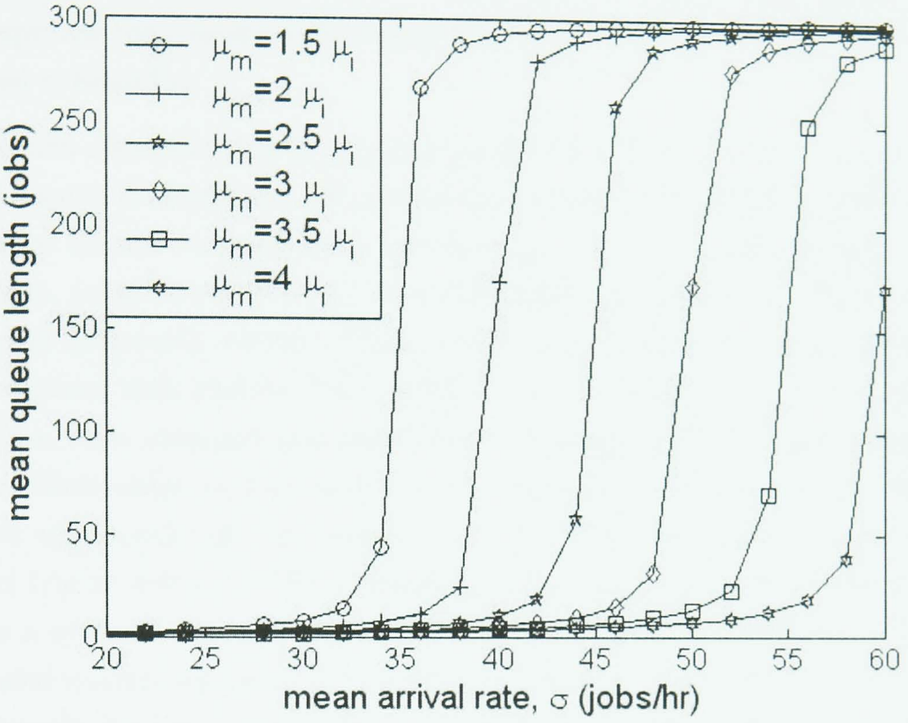

Figure 3.15: MQL versus σ for various μ_m values where L=300

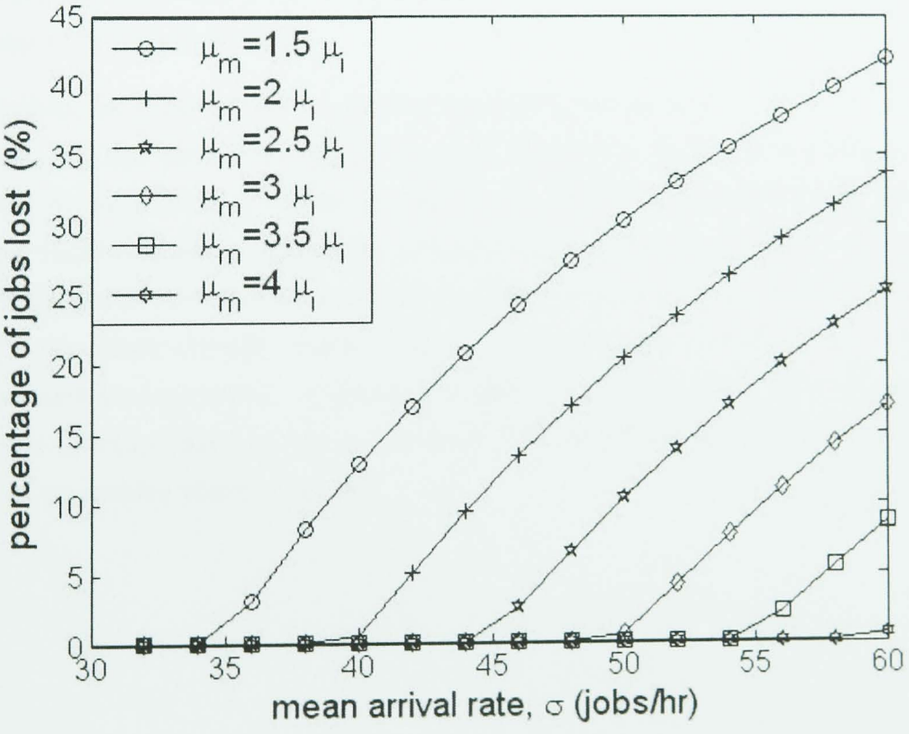

Figure 3.16: MQL versus σ for various μ_m values where L=300

In figure 3.16 the results show more clearly that increasing the service rate of the main server improves the overall performabiliy of the system significantly even when the system considered is bounded.

In this section a specific type of heterogeneous multi-server system with one main and several identical servers is considered. The operative states of the system is presented and the state probabilities for the heterogeneous multiprocessor systems with one main and several identical servers, breakdowns, repairs, reconfiguration and rebooting delays are obtained using the Spectral Expansion method. The approach presented gives a large degree of flexibility to analyse systems with various characteristics (e.g. various μ_m, μ_i, ξ_m, ξ_i values). Numerical results have been obtained and presented for various performability measures, in order to show the effectiveness off the model. Results show that, when systems with infinite queueing capacities are considered, the performability of the system highly depends on the values of $1/\delta$, and $1/\varphi$ as well as c. Also increasing the service rate and reliability of the main server improves a system's performance significantly for small scale systems. On the other hand, for bounded queuing systems, L becomes the main factor affecting the MQL performance in case of heavily loaded systems. Since the MQL value approaches to the queue size, having main servers with high service rates does not affect the system significantly. On the other hand if relatively lighter arrival rates are considered, MQL performance increases when the service rate of the main servers increases. Also when PJL performance is considered the improvement is more obvious.

In conclusion, for heterogeneous multiprocessor systems with one main and several identical processors, the discussion should go beyond whether a processor should be added or service rate of the main processor should be improved. Other parameters such as queuing capacity, and cover facilities, should also be considered in order to make an informed choice. The approach shown in this section provides the decision makers sufficient information in deciding whether a processor should be added to a system, service rate of the main processor or cover facility should be improved, or queue size should be increased when systems with one main and several identical servers are considered. The main drawback of the approach is the large number of operative states (i.e. $8K - 1$).

Chapter 4

Performability of Multi-Server Systems with One Head and Multiple Identical Servers

In order to provide the service power of multi-server systems in a user convenient way, there are various kinds of computing paradigms being effectively used. One of the most commonly used parallel computing paradigm is farm paradigms where there is a head server responsible for the overall organisation of the system. In other words the head server is the essential node of the system and the identical servers cannot provide service in case of head server breakdowns.

It is essential to use performability measures to evaluate the system, since the system considered is fault-tolerant. Furthermore, the single point of control caused by a single head architecture makes the performability analysis even more interesting and essential. The models presented in the previous chapter can be extended and QBD processes can be used for performability studies of systems using similar architectures.

Beowulf multiprocessor systems are well known examples for the architecture with one head and several identical processors which uses farm paradigm. The term, identical processor, is used for the cluster's homogeneous processors which are the main service providers to incoming jobs. Since Beowulf clusters are multiprocessor systems built from commodity off-the-shelf personal computers connected via a dedicated network and free open-source software, they can provide the high computing performance at much less or minimal cost (Wagner et. al. 1997, Adams and Vos 2002). This makes Beowulf clusters very popular as an alternative to commercial high performance systems world-wide. In this chapter Beowulf clusters are chosen as case study. Approaches are developed for performability evaluation

of Beowulf multiprocessor systems with reconfiguration and rebooting delays, by suitably modelling their operation as quasi birth and death (QBD) processes.

The approaches presented in this chapter can be used to evaluate the performability of farm paradigm systems with one head and multiple identical servers, exactly and efficiently, considering breakdowns, repairs, and delays when appropriate. Beowulf clusters can continue serving in a degraded mode provided that the failure is not at the head node. Head node failures may significantly degrade the performance of such systems. For such systems the combined evaluation of performance and availability is the most realistic and useful approach. This can provide great help for operating system design.

It is essential to understand the job handling strategies used by Beowulf systems, before modelling the system. The following section gives a brief literature review about the job handling mechanisms being used by Beowulf systems.

4.1 Job Dependencies in Beowulf Clusters

In Beowulf multi-server systems, major tasks can be splitted into a number of jobs. The tasks can be handled at the same time (in parallel) with or without communications required between the distinct pieces (jobs) and/or a controlling master process (Brown 2007).

When Beowulf systems used as computing clusters are considered, if time needed for computation is much larger than the time needed for communication ($t_{computation} \gg t_{communication}$) the application is called *coarse grained*, and in this case the computing paradigm referred to as embarrassingly parallel computing is used, where there is no communication between jobs (Brown 2004, Brown 2007). In this kind of application the computational work tends to consist of running a series of more or less independent jobs on the nodes. The user is not involved much about the design details of the cluster (Brown 2004, Fang et. al. 2005). Typical examples include the Monte Carlo calculations, statistical simulations, and data field explorations. If time needed for computation is larger than the time needed for communication ($t_{computation} > t_{communication}$) the application is called *medium grained*. In this case some communication is required between the subtasks but it is relatively small compared to the computation time of the subtasks (Brown 2007). Problems on a lattice are in this category. In some cases the computation time may be close to the communication time ($t_{computation} \approx t_{communication}$) these applications are called *less fine grained* applications (Brown 2007) and in such applications the communication time affects the overall performance of the system. Cosmology, and molecular dynamics applications with long range interactions are typical examples for this category (Brown 2004, Brown 2007).

In (Guest 2001, Grove and P.D. Coddington 2005) some benchmark results are presented for various parallel computing environments. Numerical results are presented for Beowulf cluster systems in terms of time needed for communication (time needed for message passing interface). The results indicate that the time required for communication becomes negligible compared to the computation time especially when small scale multi-server systems are used.

Beowulf multi-server systems are commonly used in high throughput applications as well. In these applications the basic aim is not to maximize the performance of each individual job by using as many processors as possible, but rather to maximize the throughput of the compute resource. In other words the aim is to provide the most efficient utilisation of the whole machine across all the jobs (Hawick et. al. 2000, Bader et. al. 2002). If the number of jobs to run is greater than the number of processors in the system, then the most efficient strategy to achieve the best throughput is to use the $job-level$ parallelism, and run each job on a separate processor (Hawick et. al. 2000). Condor is a well known example of available queuing and scheduling packages that allow a user to easily divide tasks to compute farms and to various extents balance the resource loads (Bader et.al. 2002). This package is being effectively used together with Beowulf multi-server systems (Hawick et. al. 2000, Sterling 2001). Condor supports distributed job stream resource management, emphasising capacity or throughput computing. It schedules independent jobs on cluster nodes to handle large user workloads and provides many options in scheduling policy (Sterling 2001).

Beowulf cluster systems are used in areas other than computing clusters as well such as in grid computing where job level parallelism is applied (Henty et. al. 2000, Hacker and Athey 2001, Garcia et. al. 2002, Fransisco and Akhtar 2003). Also the Open Source Cluster Application Resources (OSCAR) group developed HA-OSCAR which supports network file systems as well as web cluster facilities (Brim et. al. 2001, Haddad et. al. 2003). One goal of HA-OSCAR is to be able to work as a web server cluster providing highly available web services to a large number of clients (Haddad et. al. 2003). Both implementations are known with the use of job level parallelism.

Considering the various kinds of applications given above, it is possible to neglect the effects of job dependencies on performability since the computation process is the main factor affecting the overall performance in most of the applications. The models presented in this chapter are applicable to such systems with medium/coarse grained or high throughput applications as well as grid computing, HA-OSCAR, and small scale Beowulf systems.

4.2 Typical Beowulf Clusters with One Head and Several Computing Nodes

This section presents a model for a typical Beowulf multiprocessor system which consists of one head and multiple identical processors where the term, identical processor, is used for the cluster's homogeneous processors which are the main service providers to incoming jobs. The model is valid for systems with finite or infinite queuing capacities. The processing by the head node towards job-scheduling is negligibly small compared to the average job execution time at the identical processors (details given in the previous section). Allocation of jobs is usually done considering the availability of processors. For such a system it is well known that, in queuing theory a common queue is more suitable compared to individual queues in terms of efficiency. In addition, in Beowulf clusters, the head node may or may not participate in computations, depending on the structure of the cluster. Both of these cases are considered.

4.2.1 Modelling Typical Beowulf Clusters

The multiprocessor system considered is very similar to the one shown in figure 3.9 and it is shown in figure 4.1. The system consists of one head and $K-1$ identical parallel processors, numbered $2, 3, \ldots, K$ with a common queue. Similar to the previous systems considered, the common queue can be bounded with a capacity of L ($L \geq K$), or it can be unbounded, where jobs arrive at the system in a Poisson stream at mean arrival rate of σ, and join the queue (Hacker and Athey 2001, Yoo and Jette 2001, Gyu et. al. 2004). Jobs are homogeneous and the service rates of the identical processors are the same.

If the head processor participates in computations, usually it has the same service rate as that of the identical ones. The proposed model is applicable even if the head node's service rate is different. In some Beowulf clusters the head processor only deals with organisation and distribution of jobs and does not participate in computing (i.e., it doesn't serve).

The service times of jobs serviced by processor k ($k = 2, \ldots, K$) are distributed exponentially with mean $1/\mu_c$, and if the head node participates in computation, the service times of jobs serviced by the head processor (where $k = 1$) are distributed exponentially with mean $1/\mu_h$. Operative periods of the head processor is distributed exponentially with mean $1/\xi_h$. At the end of an operative period the head processor breaks down and requires an exponentially distributed repair time with mean $1/\eta_h$. For the identical processors ($k = 2, \ldots, K$) the mean of exponentially distributed operative periods are given by $1/\xi_c$ and the exponentially

distributed repair times have a mean value of $1/\eta_c$. The number of repairs that may proceed in parallel could be restricted. This is expressed by saying that there are R repairmen ($R \leq K$), each of whom can work on at most one repair at a time. Thus, an inoperative period of a processor would also include the possible waiting time for a repairman. In the models given in this section single repairman systems are considered, however, they can easily be adopted for the systems with multiple repairmen. Since the identical processors cannot compute if the head node is not operative, when more than one processor is broken, including the head, the repair priority is given to the head node. If the head processor is not broken the server to repair is chosen randomly. No operative processor can be idle if there are jobs awaiting service, and repairman cannot be idle if there are broken-down processors waiting for repair. All inter-arrival, service, reconfiguration, rebooting, operative and repair time random variables are distributed exponentially and are independent of each other.

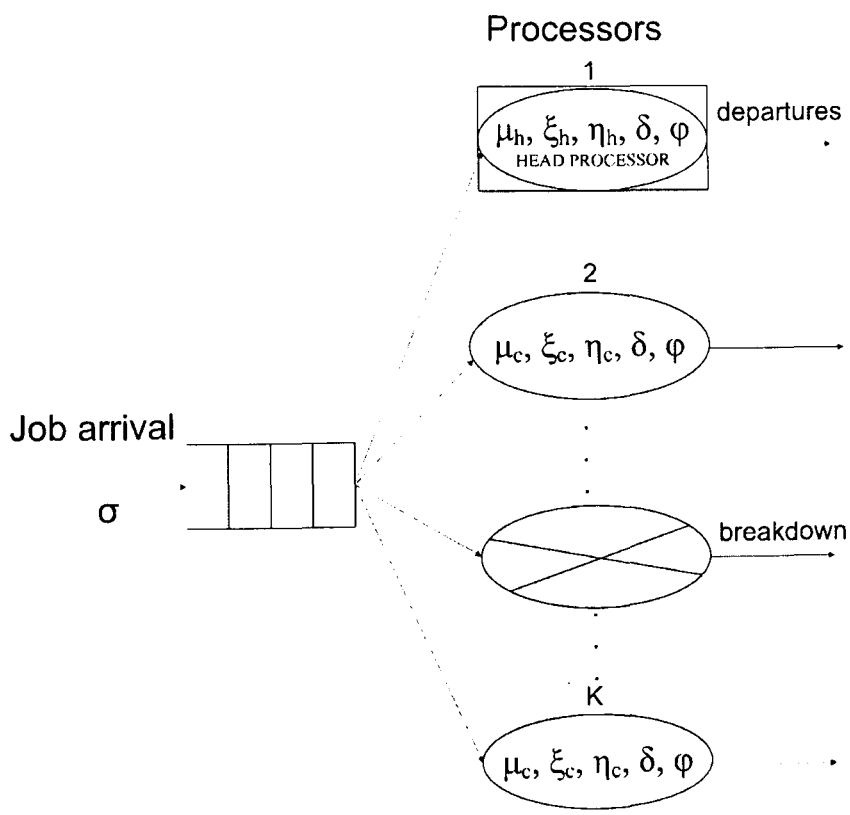

Figure 4.1: A Beowulf multiprocessor system with one head processor and $K-1$ computing processors

The reconfiguration and the rebooting delays ($1/\delta$ and $1/\varphi$) relate to the system and not to individual processors. When there are more operative processors serving the jobs, than the number of jobs in the system, the processors with higher service rates (in case the

head processor's service rate is different from the identicals) are employed. Services that are interrupted by breakdowns are eventually resumed from the point of interruption or repeated with re-sampling (perhaps on a different processor). In case of head node failures jobs continue to arrive with the same rate and, jobs in the queue remain in the queue without being serviced.

The state of the system is presented on a lattice strip. At time t random variables, $I(t)$ and $J(t)$, specify the processor configuration (can also be termed, operative state of the multiprocessor system) and the number of jobs present, respectively for Beowulf multi-server systems as well. The processor configuration, and hence the range of values of $I(t)$, refers to the available processors, (processors which are not broken) or the associated reconfiguration or rebooting states, as appropriate. Also the system's operative condition highly depends on the head processor, and healthy computing nodes are not always employed for serving (in case head is not operative).

Similar to many fault-tolerant multi-server systems (e.g. systems given in previous chapters), in Beowulf clusters some delays are encountered when a failed processor is being mapped out of the system. The reconfiguration and rebooting delays are encountered while the system is switching to an operational state, after the failure of a processor. If the head node breaks down, then, reconfiguration or rebooting is not needed since the system cannot operate without the head node. In case one of the identical processors fails and the head node is operative, the fault is covered with probability c (reconfiguration delays with probability c). Subsequent to a covered fault, the system comes up in a degraded mode after a brief reconfiguration delay, while following an uncovered fault a longer reboot action is required to bring the system up at a degraded mode (rebooting delays with probability $1-c$). During a reconfiguration/rebooting period, the system is assumed to be down (Trivedi et. al. 1990). In this section, models are developed for the typical Beowulf multiprocessor systems considered.

The Markov chain that represents all the operative states of the Beowulf cluster with a serving head node is given in figure 4.2. The state 0 means no processor is operational. The states labelled 1_h, 2_h, ..., K_h are the working states of the Beowulf multiprocessor system (i.e., head processor is operative), indicating the number of operational processors including the head processor. For instance, for $k = 1, \ldots, K$, k_h indicates that there are $k-1$ identical processors which are available alongside the head processor. The states labelled 1_c, 2_c, ..., $(K-1)_c$ are non-working states of the multiprocessor system, where the head processor is not operative and the number indicates the number of available identical processors. The states, labelled as X_1, X_2, ..., $X_{(K-1)}$, and Y_1, Y_2, ..., $Y_{(K-1)}$ are the states representing the reconfiguration and rebooting delays respectively when a computing processor fails but the

head processor is operative.

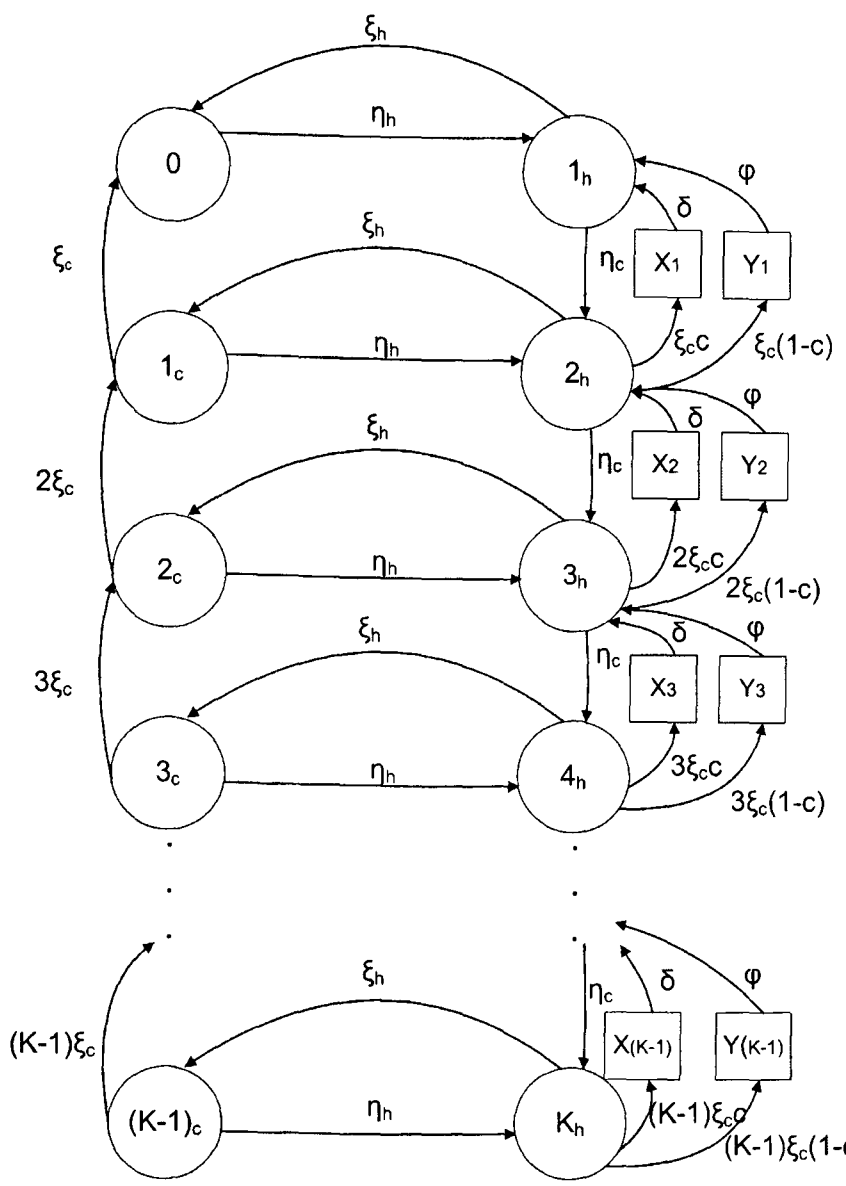

Figure 4.2: The operative states of a typical Beowulf multiprocessor system with a serving head node

A similar model can be given for the systems with a non-serving head node. Figure 4.3 shows the operative states of such a system. The only difference between figure 4.2 and 4.3 is that, in the latter, reconfiguration or rebooting delays are not presented when transitions from state 2_h to state 1_h occur. This is because the system does not provide any service in state 1_h, since the head node does not serve incoming jobs. The notation used in Figure 4.2 is used for Figure 4.3 as well, but the reconfiguration/rebooting stages X_1, and Y_1 do not exist.

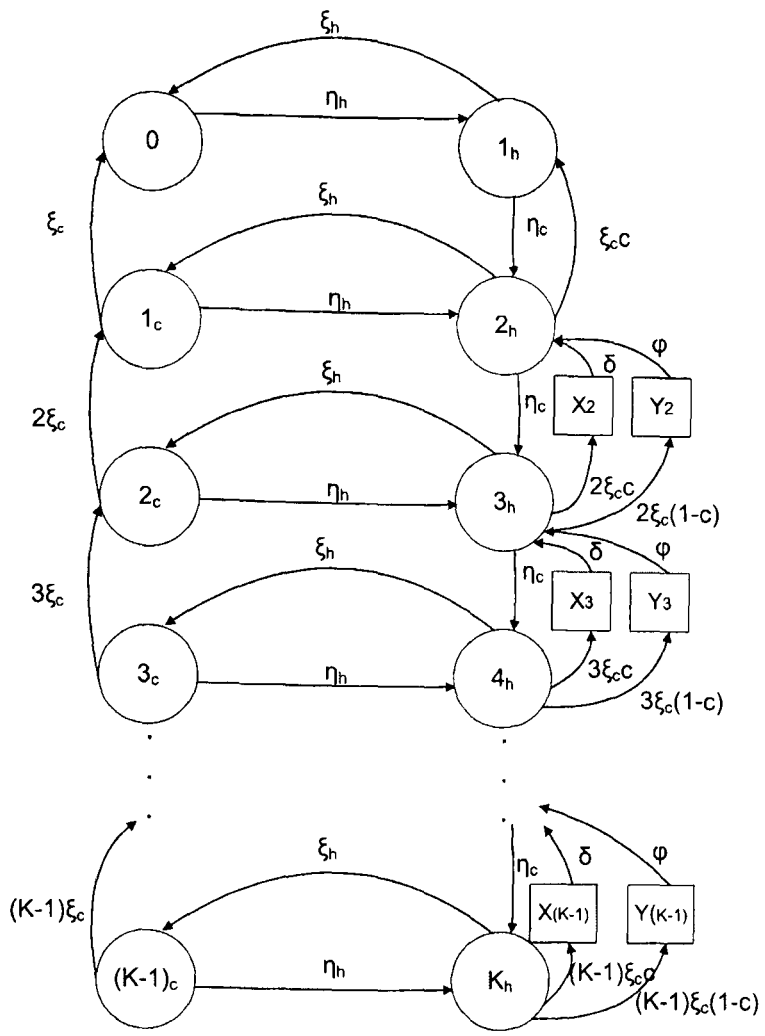

Figure 4.3: The operative states of a typical Beowulf multiprocessor system with a non-serving head node

The systems with K processors, are presented using a QBD process with finite or infinite state space. The elements of matrix A depend on the parameters K, ξ_h, ξ_c, η_h, η_c, c, δ and φ. Matrix B is a diagonal matrix with arrival rate on the main diagonal. Matrix C is another diagonal matrix which has the service rates provided appropriately for each operative state on the main diagonal. The total number of states for the system with a serving head-node is given by $4K-2$. Numbering of these operative states can be arbitrary. For systems with a serving head-node, we assign odd numbers $(1,3,5,\ldots,2K-1)$ to the states (1_h, 2_h ,..., K_h) respectively, and, zero and even numbers $(0,2,4,\ldots,2K-2)$ to the states (0, 1_c, 2_c, ..., $(K-1)_c$). The reconfiguration states X_1, X_2 , ..., $X_{(K-1)}$ are represented by $(2K, 2K+1,\ldots,3K-2)$ and the rebooting states Y_1 , Y_2 , ..., $Y_{(K-1)}$ are given the numbers $(3K-1, 3K,\ldots,4K-3)$. This is a total of $4K-2$ states.

For a K-processor Beowulf cluster, with a serving head node the matrices B, and C can be

given as follows:

$B_j = B$ for $j = 0, 1, \ldots, L$; $B = \text{Diag}[\sigma, \sigma, \ldots, \sigma]$ of size $(4K-2) \times (4K-2)$

$C_j = C$ for $j \geq K$; the threshold $M = K$
$C = \text{Diag}[0, \mu_h, 0, (\mu_h + \mu_c), 0, (\mu_h + 2c), 0, \ldots, \mu_h + (K-1)\mu_c), 0, \ldots, 0]$
$C_0 = 0$.
If $\mu_h > \mu_c$, the head node has service priority over the identical ones. This means that, when the system is operative the head node provides service and the remaining jobs (if any) go to the identical computing nodes.

Hence, $C_j = \text{Diag}[0,\ (\mu_h + Min\{w_c(1), j-1\}\mu_c),\ 0,\ (\mu_h + Min\{w_c(3), j-1\}\mu_c),\ 0,\ \ldots,$ $(\mu_h + Min\{w_c(2K-1), j-1\}\mu_c),\ 0,\ \ldots,\ 0]$ for $1 \leq j < K$.

If $\mu_h \leq \mu_c$, then, the identical nodes have service priority over the head. In this case, when the system is operative the jobs first go to the identical computing nodes. If the number of jobs is greater than the number of available identical nodes then head processor is used to serve one of the jobs.

Hence, $Cj = \text{Diag}[0,\ (Min\{w_c(1), j\}\mu_c + Min\{1, j - Min\{w_c(1), j\}\}\mu_h),\ 0,\ (Min\{w_c(3), j\}\mu_c + Min\{1, j - Min\{w_c(3), j\}\}\mu_h),\ 0,\ \ldots,\ (Min\{w_c(2K-1), j\}\mu_c + Min\{1, j - Min\{w_c(2K-1), j\}\}\mu_h)\ ,\ 0,\ \ldots,\ 0]$ for $1 \leq j < K$.

where $w_c(i)$ is the number of identical processors available in the operative state i. Here, $Min\{w_c(k), j-1\}$ means minimum of the number of operative identical servers in state k and the number of jobs (since one job is serviced by head node $j-1$ is taken). $Min\{1, j - Min\{w_c(k), j\}\}$ means minimum of the number of head nodes (one since there is a single head node) and the number of jobs remaining after all of the available identical computers are engaged in state k.

Since the system can only work when the head processor is operational, there is no service during even numbered states (i.e., $i = 0,\ 2,\ \ldots,\ 2K-2$). The final $2K-2$ diagonal elements of matrices C and C_j are always zero since they are used to represent the reconfiguration/rebooting states. The matrix A is too bulky to be written in compact form, as an example, for $K = 4$, A can be written as follows:

$$A_j = A = \begin{pmatrix}
0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_h & 0 & 0 & \eta_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_c & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \xi_h & 0 & 0 & \eta_c & 0 & 0 & \xi_c c & 0 & 0 & \xi_c(1-c) & 0 & 0 \\
0 & 0 & 2\xi_c & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \xi_h & 0 & 0 & \eta_c & 0 & 2\xi_c c & 0 & 0 & 2\xi_c(1-c) & 0 \\
0 & 0 & 0 & 0 & 3\xi_c & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 & 0 & 0 & 3\xi_c c & 0 & 0 & 3\xi_c(1-c) \\
0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Similarly the total number of states for the system with a non-serving head node is given by $4K-4$. The odd numbers $(1, 3, 5, \ldots, 2K-1)$ are assigned to the states (1_h, 2_h ,..., K_h) respectively, and, zero and even numbers $(0, 2, 4, \ldots, 2K-2)$ to the states (0,1_c, 2_c ,..., $(K-1)_c$). The reconfiguration states X_2, X_3, ..., $X_{(K-1)}$ are represented by $(2K, 2K+1, \ldots, 3K-3)$ and the rebooting states Y_2, Y_3, ..., $Y_{(K-1)}$ are given the numbers $(3K-2, 3K-1, \ldots, 4K-5)$.

The matrices B,and C for a K-processor Beowulf cluster with a non-serving head node, can be given as follows:

$B_j = B$ for $j = 0, 1, \ldots, L$; $B = \text{Diag}[\sigma, \sigma, \ldots, \sigma]$ of size $(4K-4) \times (4K-4)$

$C_j = C$ for $j \geq K$; the threshold $M = K$
$C = \text{Diag}[0, 0, 0, \mu_c, 0, 2\mu_c, 0, \ldots, (K-1)\mu_c, 0, 0, \ldots, 0]$
$C_0 = 0.$
$C_j = \text{Diag}[0, 0, 0, (Min\{w_c(3), j\}\mu_c), 0, (Min\{w_c(5), j\}\mu_c), 0, \ldots, (Min\{w_c(2K-1), j\}\mu_c), 0, \ldots, 0]$
for $1 \leq j < K$.

For this system as well, there is no service during the even-numbered states (i.e., $i = 0$, 2, ..., $2K-2$) since the system can work only if the head processor is operational. Also in this system, since the main processor does not serve, there is no service at state 1_h. The last $2K-4$ diagonal elements of matrices C and C_j are always 0, since they are used to represent the reconfiguration/rebooting states. For a system with four servers ($K = 4$), the matrix A becomes:

$$A_j = A = \begin{pmatrix}
0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_h & 0 & 0 & \eta_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_c & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \xi_c & \xi_h & 0 & 0 & \eta_c & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2\xi_c & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \xi_h & 0 & 0 & \eta_c & 2\xi_c c & 0 & 2\xi_c(1-c) & 0 \\
0 & 0 & 0 & 0 & 3\xi_c & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 & 0 & 3\xi_c c & 0 & 3\xi_c(1-c) \\
0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Once the matrices are prepared the Spectral Expansion method is employed to solve the system for state probabilities.

4.2.2 Numerical Results and Discussions for Typical Beowulf Systems

Numerical results are presented in this section to show the effectiveness of the models developed for typical Beowulf clusters, and to evaluate the performance of these systems. The parameters used in this numerical study are mainly taken for systems with highly reliable components similar to the studies given in (Trivedi et. al. 1990, Leangsuksun et. al. 2004). However, the models provide large degrees of flexibility and it is possible to obtain numerical results for systems with various diverse characteristics. Beowulf multiprocessor systems with breakdowns and infinite queues are considered first.

Beowulf Clusters with Unbounded Queues

This section presents results on non-blocking Beowulf Clusters (system with an unbounded queue).

In figure 4.4, Beowulf clusters with 2, 3, 4 and 8-processors, with computing and non-computing head processors, are considered. Other parameters are given as σ jobs/hour, $\xi_h = \xi_c = (1/6000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\mu_h = \mu_c = 300$ jobs/hr, $\varphi = 12$/hr (rebooting delay is 5 minutes), and $\delta = 120$ /hr (reconfiguration delay is 30 seconds). This figure shows the relationship between MQL and the mean arrival rate , for different numbers of servers and $c = 0$. Figure 4.4 shows that systems with greater number of servers perform better as expected ((a) shows loaded systems, (b) shows systems with relatively lighter loads). If the systems with the same number of serving processors, (e.g. $K = 4$ when head node is not

serving and $K = 3$ when head node serving) are considered, systems with serving head nodes perform better. This is because for the systems with non-serving head nodes the number of states where the system is not operational is greater.

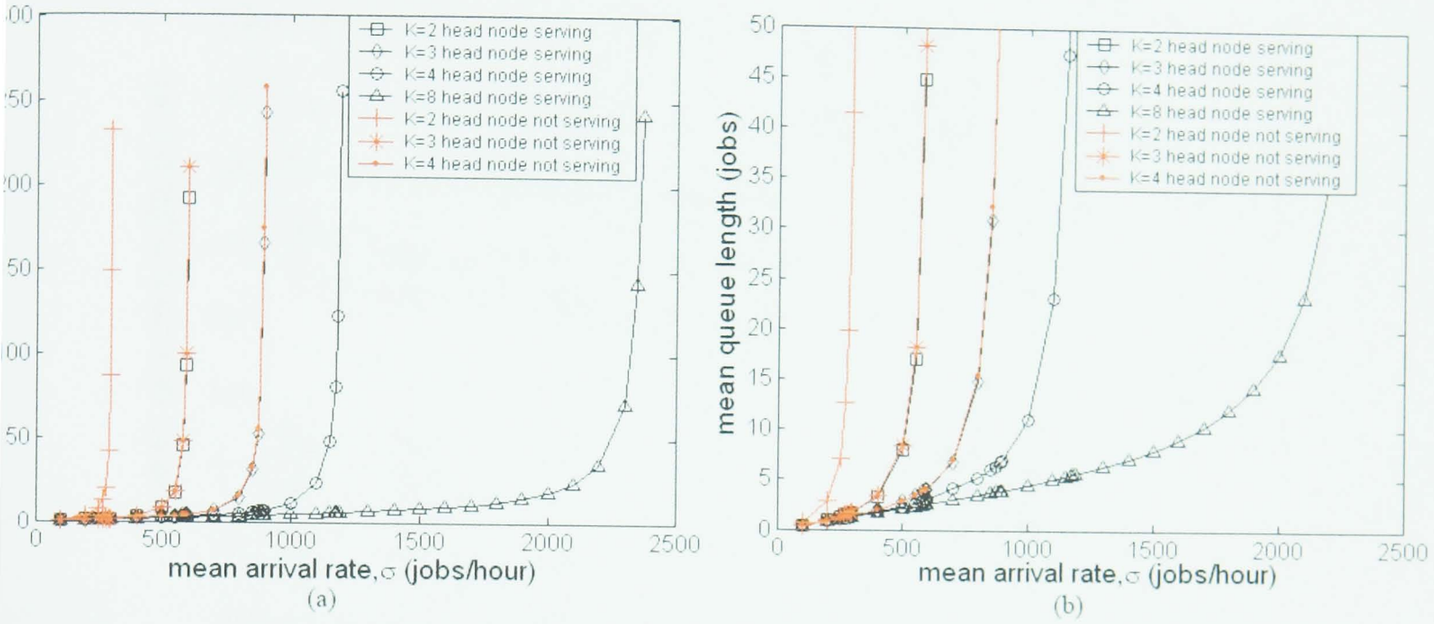

Figure 4.4: MQL versus σ for Beowulf systems with various K

For a system with K nodes, the number of states where the system is not providing service can be obtained by subtracting the number of states where the system is serving from the total number of nodes. For a system with K nodes where the head node does not serve, the total number of states where the system is not serving can be given as $(4K-4)-(K-1) = 3K-3$. For the systems with a serving head node the total number of states where the system is not operative is $(4K-2)-(K) = 3K-2$. Let's consider two systems with equal numbers of serving nodes: For a system with four nodes including a non-serving head node there are nine states where the system does not provide service. On the other hand, for a system with three nodes, including a serving head node, there are seven states where the system does not provide service. This gives rise to better performance in the case of the latter.

The parameters $1/\delta$ and $1/\varphi$ play an important role in determining a Beowulf system's performance degradation due to reconfiguration/rebooting delays for multiprocessor systems. The mean queue length has been computed as a function of reconfiguration rate δ for systems with $K = 3$. Other parameters are $\sigma = 3$ job/sec, $\sigma/(K\mu_h) = 0.7$ (i.e. 70% utilisation

if there were no breakdowns, constant utilisation is used to analyse efficiency), $\mu_c = \mu_h$, $\xi_h = \xi_c = (1/1000)$/hr, $\eta_h = \eta_c = 0.5$/hr, and $\varphi = 3$/hr. The results are illustrated in Figure 4.5.

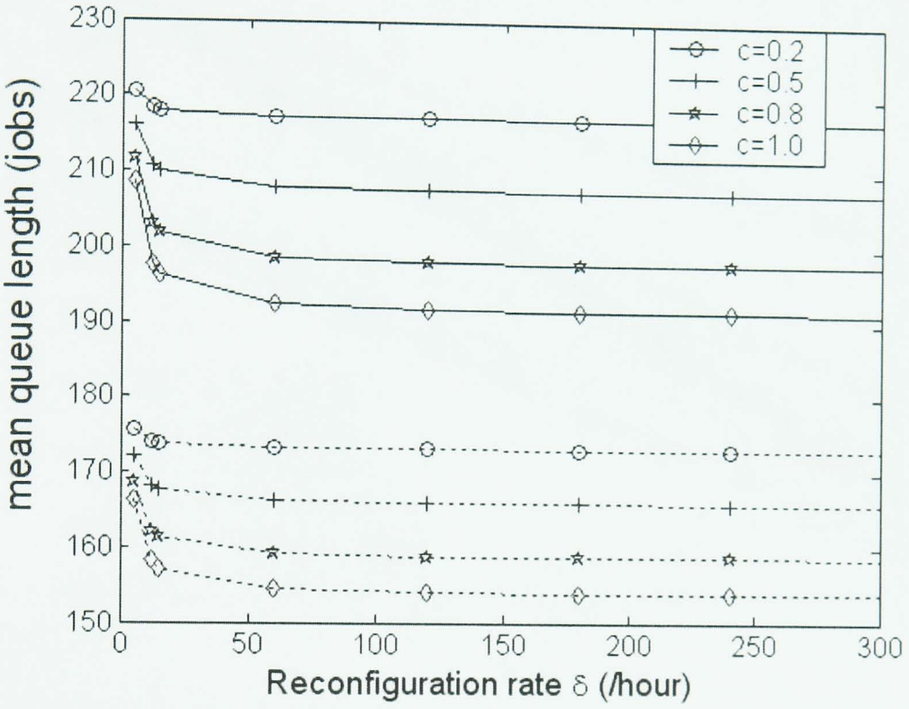

Figure 4.5: MQL as a function of c and δ for $K = 3$ ($\cdots$ serving head node, — non-serving head node)

Results show how MQL changes as reconfiguration delay ($1/\delta$) decreases. Since the parameters other than reconfiguration delay (e.g. rebooting delay, failures and repairs) become more significant at higher reconfiguration delays (i.e. for $1/\delta > 1$ minutes which means $\delta > 60$ per hour), the changes in MQL performance become less significant. Also this figure shows that systems with one extra serving node (serving head node) perform relatively better in case of systems with three nodes.

Figure 4.6 shows how mean queue length decreases as c increases for systems with 3 serving nodes (i.e., $K = 4$ when head node not serving and $K = 3$ when head node serving). The parameters used are $\sigma = 3$ jobs/sec, $\sigma/(K\mu_h) = 0.7$, $\mu_c = \mu_h$, $\xi_h = \xi_c = (1/1000)$/hr, $\eta_h = \eta_c = 0.5$/hr, and $\varphi = 3$ /hr. Results are obtained for various values of reconfiguration delay $1/\delta$. For larger reconfiguration delays performance degradation is evident even at higher c values. As the reconfiguration delay decreases the degradation is mainly due to c

values. Similar to figure 4.4 this figure also shows that systems with a serving head node perform better when the number of nodes providing service is same in total.

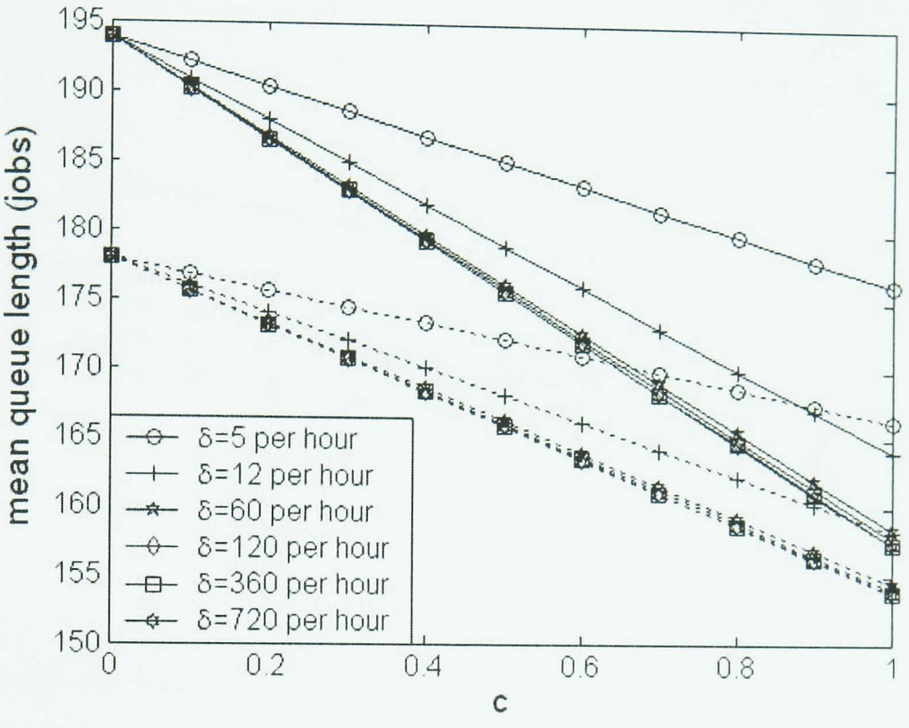

Figure 4.6: MQL as a function of c and δ ($\cdots$ serving head node, — non-serving head node)

Figure 4.7 shows mean queue length as a function of c, with $K = 2$, 3 and 4, $\sigma = 3$ jobs/sec, $\sigma/(K\mu_h) = 0.7$, $\mu_c = \mu_h$, $\xi_h = \xi_c = (1/1000)/\text{hr}$, $\eta_h = \eta_c = 0.5/\text{hr}$, $\varphi = 3/\text{hr}$, and $\delta = 120/\text{hr}$ for systems with non-serving and serving head nodes. Results show that for systems with a non-serving head, the 3 and 4-processor systems can have relatively close mean queue length performances. This is mainly because of the service rates which depend on the total number of servers in the system. Please recall that $\sigma/(K\mu_h) = 0.7$. Furthermore, the performances of two processor system with a serving head and four processor system with a non-serving head are very close, the latter becoming better for $c > 0.5$. This shows that if good cover cannot be provided, the use of four processors will not be efficient ($\sigma/(K\mu_h) = 0.7$ constant utilisation is used to analyse efficiency).

Similar computations are provided in figure 4.8. The mean queue length is again presented as a function of c. This time, service rates of head and identical processors are taken constant as $\mu_h = \mu_c = 6000$ jobs/hr.

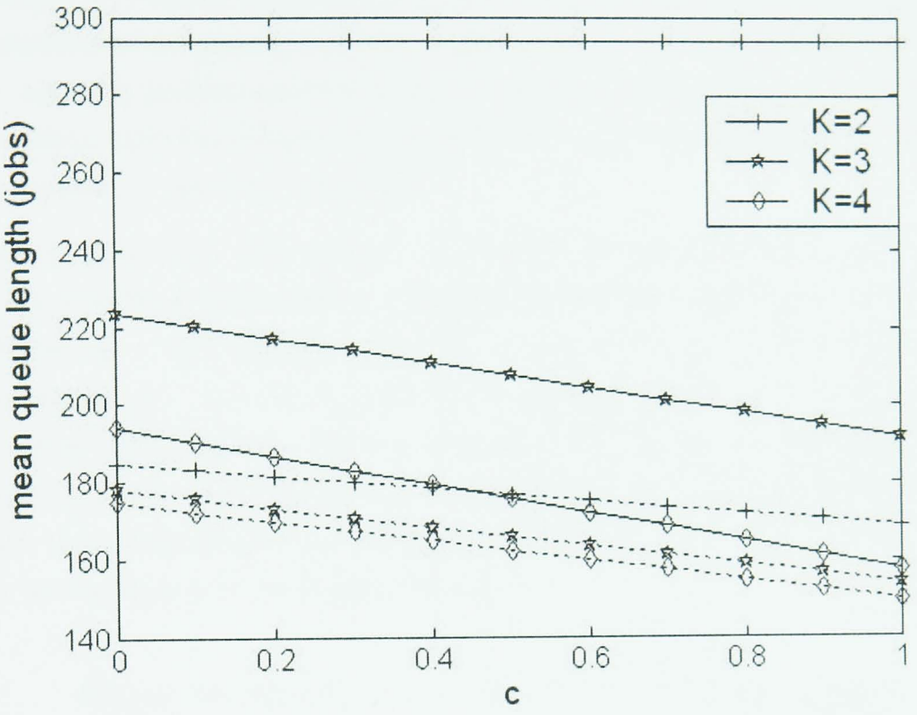

Figure 4.7: MQL as a function of c for various K where $\sigma/(K\mu_h) = 0.7$ ($\cdots$ serving head node, — non-serving head node)

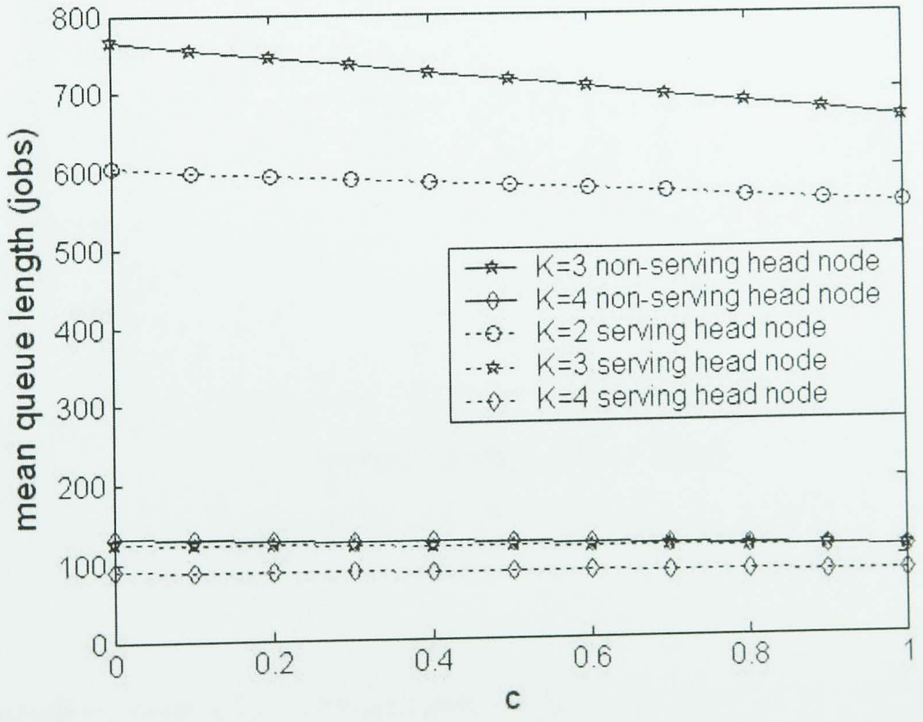

Figure 4.8: MQL as a function of c for various K where $\mu_c = \mu_h = 6000$ jobs/hr

Unlike figure 4.7, figure 4.8 shows that when service rates are constant, the systems with greater number of processors perform substantially better in typical Beowulf multiprocessor systems, also the performances of four processor systems with non serving head node, and three processor systems with serving head node (i.e. same number of serving nodes) are close since the system is not heavily loaded.

In figure 4.9, the MQL performance of Beowulf Clusters with 2, 4, and 8-processors with serving head nodes are considered. Other parameters are given as $\sigma = 3$ jobs/sec, $\xi_h = \xi_c =$ (1/1000)/hr, $\eta_h = \eta_c = 0.5$/hr, $\sigma/(K\mu_h) = 0.7$, $c = 1.0$, $\mu_c = \mu_h$. Results clearly show that, for $\sigma/(K\mu_h) = 0.7$, K=4 performs marginally better than K=8 making 4 processor systems more cost effective. Hence, increasing number of computing nodes may result in less efficiency due to increased loss of computation power caused by breakdowns and delays. Here with these parameters two independent $K/2$ processor systems with equal load sharing perform better than a K processor system.

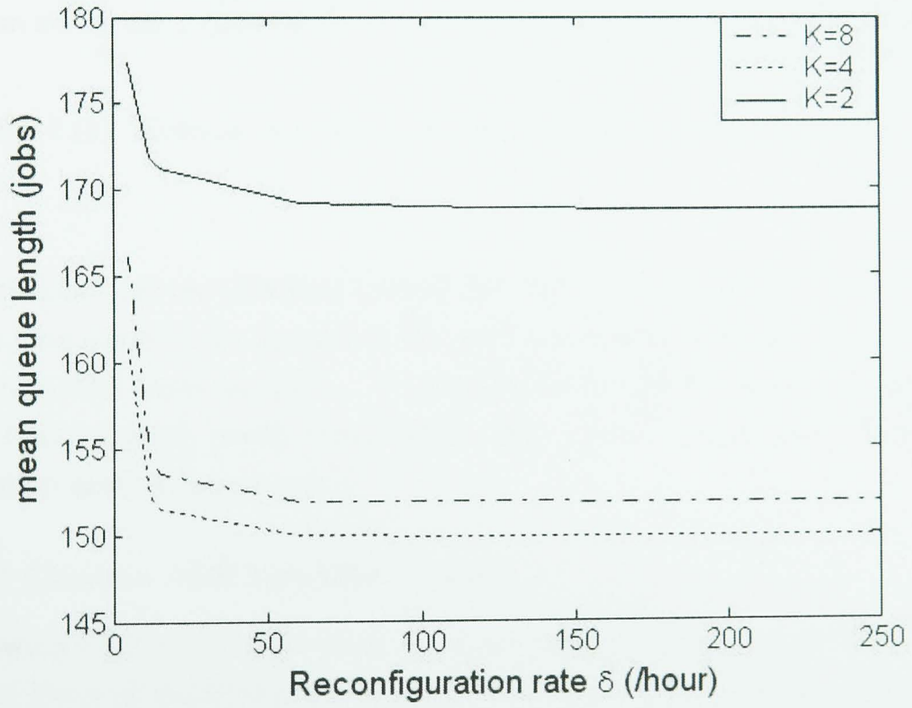

Figure 4.9: MQL as a function of δ and K

The parameters used in figure 4.4 are used for figure 4.10 as well. The average response times of typical Beowulf clusters with various number of processors and a serving head node are presented ((a) shows loaded systems, (b) shows systems with relatively lighter loads). This

figure shows that similar to figure 4.4, the average response time decreases as the number of processors increases, as expected.

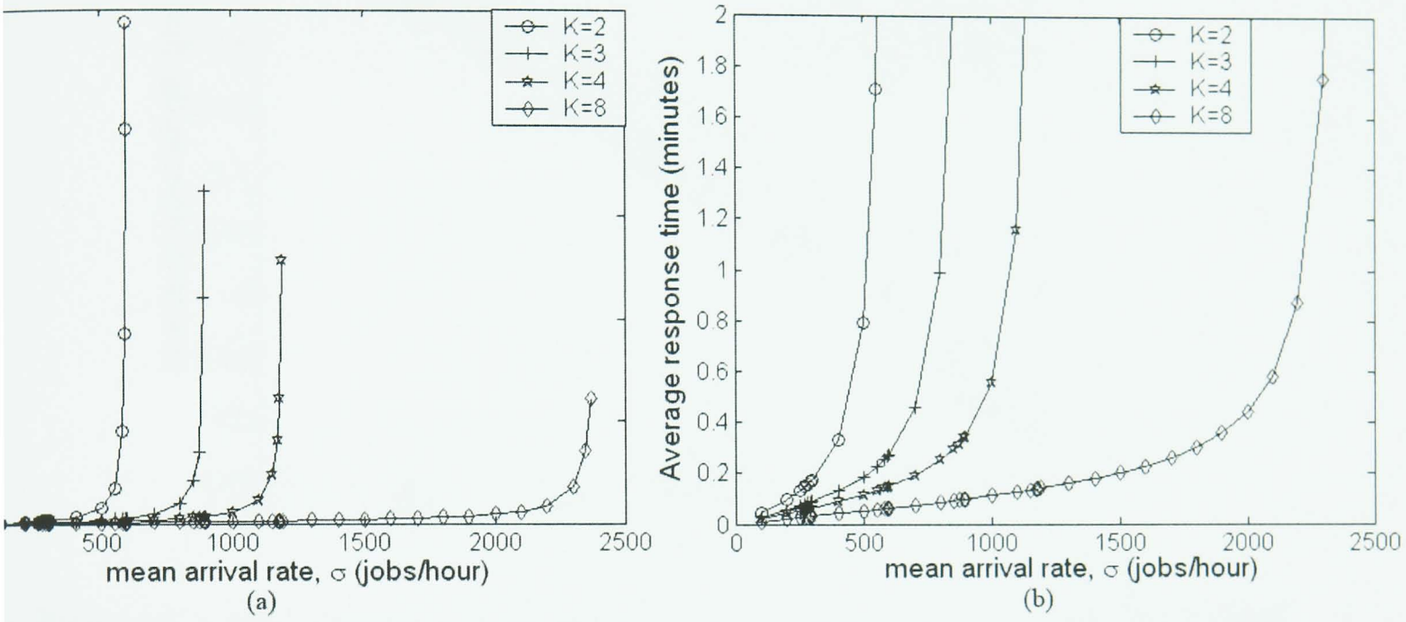

Figure 4.10: Average response time as a function of σ and K for Beowulf systems

Results obtained for non-blocking typical Beowulf systems show that, having a serving head node can have a significant impact on the system's performance even if the number of serving processors is the same in total. The regularity of graphs (smooth transitions including straight lines in some cases) means that these curves can be used to predict MQLs by interpolation and regression easily.

Beowulf Clusters with Bounded Queues

The following results have been obtained to demonstrate the effects of finite queuing capacity on typical Beowulf multiprocessor systems with one head, and several identical processors. Figure 4.11 shows how mean queue length decreases as c increases for a system with bounded queuing capacity, two identical nodes and a serving head node. Other parameters are $\sigma = 120$ jobs/hour, $\sigma/(K\mu_h) = 0.7$, $\mu_c = \mu_h$, $\xi_h = \xi_c = (1/1000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\varphi = 2$/hr, and $L = 1000$. Results are obtained for various values of δ.

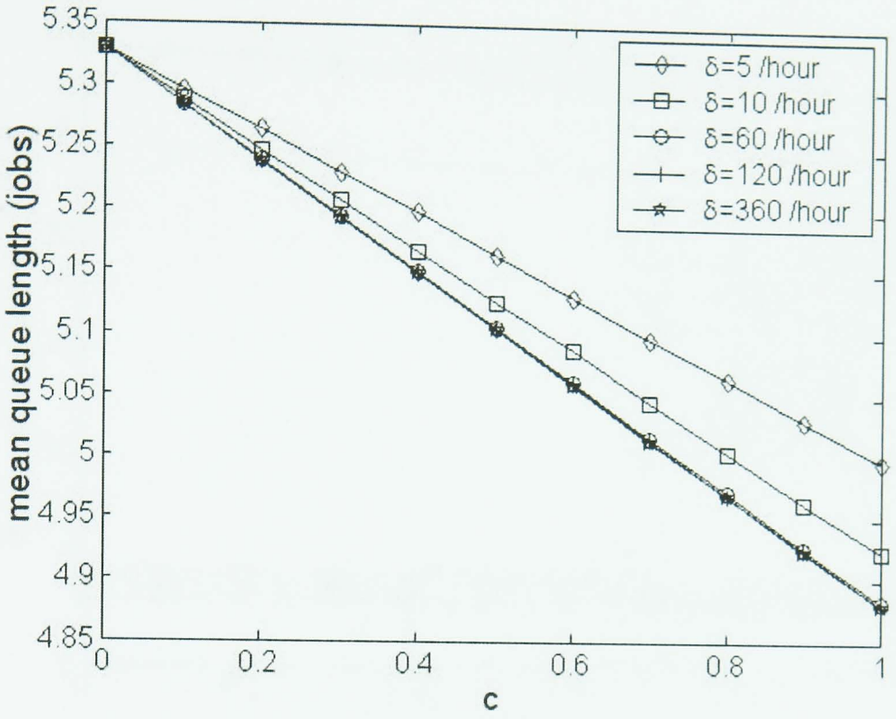

Figure 4.11: MQL as a function of c and δ for a system with 2 identical and a serving head node, and L= 100

Here, since $\sigma/(K\mu_h) = 0.7$ and the system is bounded, reconfiguration delays and c do not affect the system's MQL performance as significantly as unbounded systems. Also similar to the unbounded systems decreasing the reconfiguration delay (i.e. increasing δ) does not affect the system after a certain value ($\delta = 60$ here)since the other factors become more significant.

Figure 4.12 shows the mean queue length as a function of c, with $K = 2$, 3 and 4, $\sigma = 150$ jobs/hour, $\mu_h = \mu_c = 300$ jobs/hr, $\xi_h = \xi_c = (1/1000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\varphi = 2$/hr, and $\delta = 360$/hr. Systems with a serving head node are considered. Results show that for systems with relatively smaller number of servers (e.g., $K = 2$) the queuing capacity affects the mean queue length performance more significantly. This is mainly because of the relatively small σ value used. Also the figure shows that even though the rebooting delays are much larger than reconfiguration delays, the effect of c on mean queue length performance is not substantial.

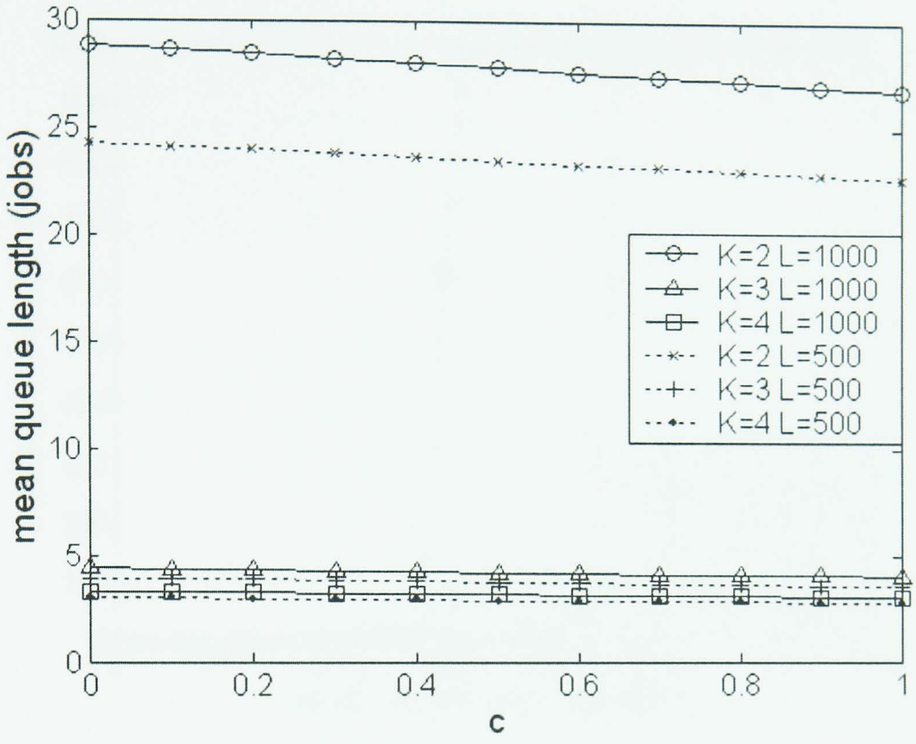

Figure 4.12: MQL as a function of K, and c for bounded Beowulf systems

In Figures 4.13, and 4.14 the mean queue length is calculated for systems with serving and non serving head nodes respectively. The other parameters are $K = 3$ and 4, $\mu_h = \mu_c = 300$ jobs/hr, $\xi_h = \xi_c = (1/1000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\varphi = 3$/hr, $\delta = 120$/hr, and $L = 1000$.

The results show that changes in c values do not affect the system performance and L is the main limiting factor in case of Beowulf clusters with finite queuing capacities. For relatively small arrival rates, systems with greater number of processors perform better, but in case of large arrival rates again L becomes the main limiting factor. When the figures 4.13 and 4.14 are compared it is also evident that systems with same number of serving nodes (K=4 for systems with a non-serving head node, and K=3 systems with a serving head node) have very close MQL performances. Because of the limiting effect of L systems with serving head nodes do not perform better.

Figures 4.15 and 4.16 are used to show these results by using the percentage of jobs lost in typical Beowulf multiprocessor systems, due to buffer saturation. These results show that systems with same number of serving nodes have very close PJL performances as well.

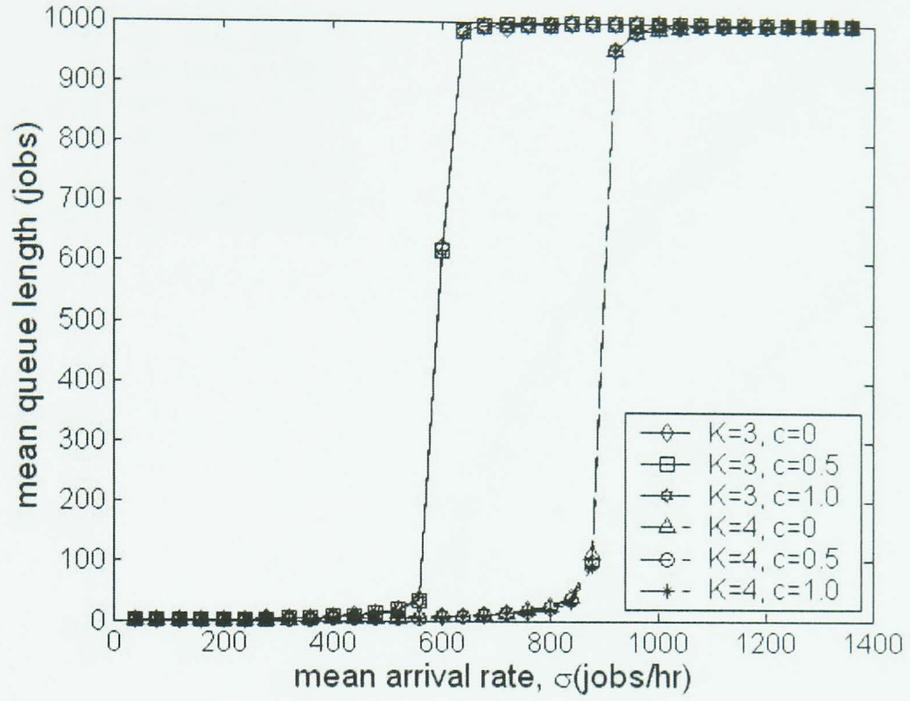

Figure 4.13: MQL as a function of K, and σ for systems with a non-serving head node and $L = 1000$

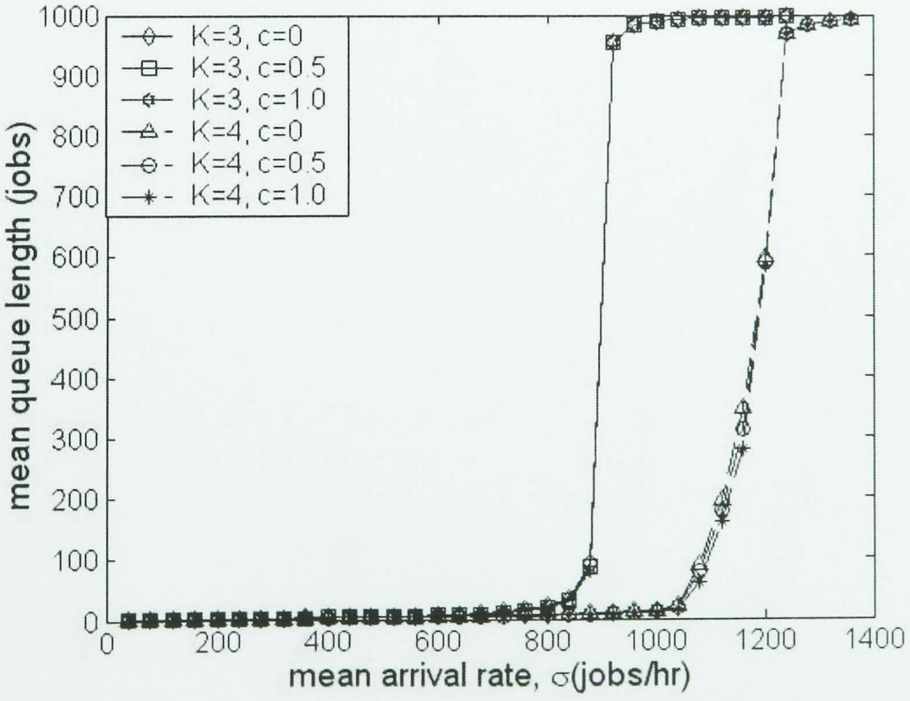

Figure 4.14: MQL as a function of K, and σ for systems with a serving head node and $L = 1000$

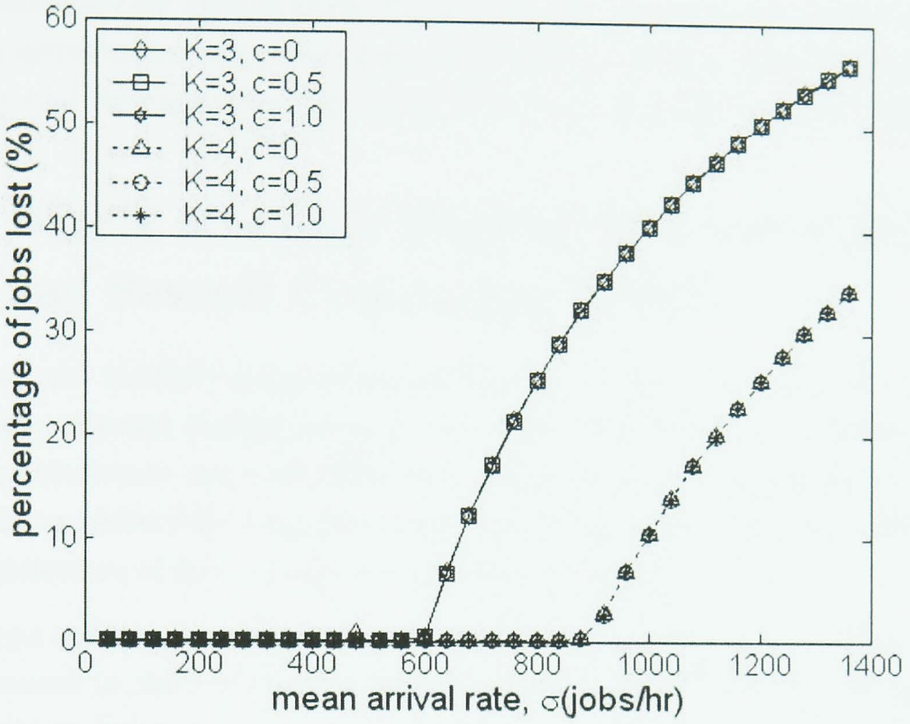

Figure 4.15: PJL as a function of K, and σ for systems with a non-serving head node and $L = 1000$

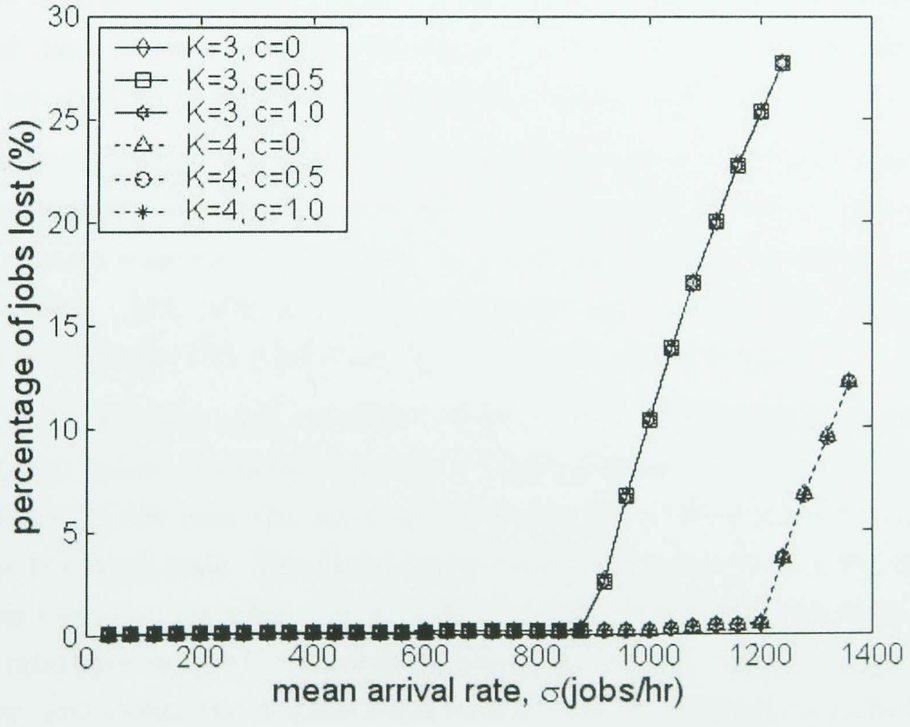

Figure 4.16: PJL as a function of K, and σ for systems with a serving head node and $L = 1000$

Results obtained for typical Beowulf systems with bounded queuing capacities show that, for large arrival rates, the delays and cover facilities do not affect system performance significantly and the capacity of the queue is the main limiting factor for loaded networks.

4.3 Highly Available Beowulf Clusters with One Head and Several Computing Nodes

One head and multiple compute nodes structure is used generally for many typical high performance clusters (Leangsuksun et. al. 2004). Since the most pressing issues of today's cluster architectures are availability and serviceability (Leangsuksun at. al. 2005), it is important to address the relatively weak availability that may be caused by the single head node architecture of farm paradigm multi-server systems.

Providing a spear head server for replacement in case of head node breakdowns is a commonly used approach in order to increase the availability. Cold and hot standby techniques are two well known techniques which are used in order to provide backup systems in case of head node breakdowns.

For cold standby replacement technique the operator has a configured backup system and a plan to recover the services of the system. The systems which use cold standby, receive scheduled data backups, but this process is less frequent than hot-standby. Cold standby systems are used for non-critical applications or in cases where data is changed infrequently.

Hot standby technique is a method of replacement where the primary and backup systems run simultaneously. The data is mirrored to the secondary server in real time so that both systems contain identical information. This technique allows for minimal downtime in case of breakdowns. The switching delay is shorter than cold standby switching delay. The applications that can risk a minimal down time use this technique.

HA-OSCAR is the first highly available (HA) Beowulf cluster that provides high availability and a critical failure prediction capability (Leangsuksun et. al. 2003, Leangsuksun et. al. 2004). HA-OSCAR uses the hot standby technique to deal with the single point failure caused by the head node. The hot-standby head node is a mirror of the original head node and serves user requests when the original head node fails. In case of head node failures, the network interfaces of head node drop (Leangsuksun et. al. 2004). The standby head node takes over, and clones the original head node's network configuration. This operation takes three to five seconds (Leangsuksun et. al. 2003, Leangsuksun et. al. 2004).

In this section, models are developed for performability evaluation of highly available Beowulf

like high performance clusters. The model is valid for systems which use hot and cold standby techniques. Some assumptions are made in order to reduce the number of states for a system with K nodes. The assumptions are stated and validated. It is known that the states where the original head node is active and the states where the standby head node is active provide the same service rates (i.e., in case of head node failures the service rate of the system remains the same after the standby processor takes over the control).

4.3.1 Modelling Highly Available Beowulf Clusters

The highly available multi-server system considered consists of one head (numbered 1), and $K-1$ identical parallel servers, (numbered 2, 3, ..., K), and a standby backup processor for the head processor. For highly available systems as well, a common queue is considered which can be bounded with a capacity of L ($L \geq K$), or unbounded. Jobs arrive at the system in a Poisson stream (Hacker and Athey 2001, Gyu et. al. 2004) at mean rate of σ, and join the queue. Jobs are homogeneous and the service rates of the identical processors are the same. The head node may or may not provide service to the incoming job requests. In case of systems where the head processor participates in the computations, the mean service rate of the head node is usually the same as the mean service rate of the identical ones. The proposed model is applicable even if the head node's mean service rate is different.

The failure rate of the original and the standby head nodes are given as ξ_h and ξ_s respectively. The failure rate of the computing nodes (identical ones) is ξ_c. At the end of an operative period, one of the computing processors breaks down and requires an exponentially distributed repair time with mean $1/\eta_c$. On the other hand, if the original head processor or the standby head processor breaks down, the repair facility is provided with mean repair time $1/\eta_h$, and $1/\eta_s$ respectively. The service rates of the identical processors are given as μ_c. Service rates of the original and standby head processors are equal and they are given as μ_h and μ_s respectively. The time needed for standby head processor to take over the control in case of original head processor failures is called the *switching delay*. The reconfiguration delay $1/\delta$ rebooting delay $1/\varphi$ and the switching delay $1/\zeta$ relate to the system and not to individual processors.

Once the original head processor fails, the standby processor takes over the control. The repair facility is called as soon as the head node breaks down in order to provide high availability. Since the repair time of the processors are much shorter than the failure time of the processors, the original head node is repaired before the standby head node breaks down. In other words since $1/\eta_h \ll 1/\xi_s$ (the mean repair time of original head processor is much smaller than the average time-to-fail of the standby head processor), it is possible

to assume that the original head processor is repaired before a failure occurs at the standby head processor.

In such a system the head node failures affect the systems performability only because of the switching delays. The states with original head processor, and standby head processors can be merged and represented with a single state, since the original and standby head processors have the same service rates (they are mirrors of each other). Two separate computations have been performed to compute the mean queue length values of a four processor system with serving head nodes. The non-merged model was considered as well as the merged model. The parameters are taken as $c = 0$, σ jobs/second, $\xi_h = \xi_s = \xi_c = 0.01$/hr, $\eta_h = \eta_s = \eta_c = 0.5$/hr, $\mu_h = \mu_s = \mu_c = 8000$ jobs/hr, $\varphi = 2$/hr, and $\zeta = \delta = 60$/hr.

Table 4.1: Comparison between merged and non-merged models

σ	*MQL non-merged model*	*MQL merged model*
1	42.077587	42.077309
1.2	57.604926	57.604528
1.4	77.704696	77.704124
1.6	104.441595	104.440755
1.8	141.30543	141.304145
2.0	194.668833	194.666735
2.2	277.397744	277.394067
2.4	397.28126	397.276232
2.6	545.735976	545.729152
2.8	733.443164	733.433703
3.0	978.344166	978.330646

The results given in table 4.1 show that although the failure rate is taken relatively high (i.e., $\xi_h = \xi_c = 0.01$/hr, $\eta_h = \eta_c = 0.5$/hr), the differences of computed MQL values were less than 0.002% which is negligibly small. Hence, the use of the merged model which gives smaller number of states is well justified.

Since the original and standby head processors are the mirrors of each other, they have the same characteristics. In the following parts of this section the failure, repair, and service rates of both original and standby head nodes are noted as ξ_h, η_h, and μ_h respectively.

The model developed for highly available Beowulf clusters with a serving head node is given in figure 4.17. The states labelled 1, 2, ..., K are the working states of the multiprocessor system (i.e., original or the standby head processor is operative), indicating the number of operational processors including the head processor. The states, labelled as X_1, X_2, ...,

$X_{(K-1)}$, and Y_1, Y_2, ..., $Y_{(K-1)}$ are the states representing the reconfiguration and rebooting delays respectively. In case of failure at the original head processor the standby processor takes over the control after switching delay. Systems which use a cold standby technique have relatively larger switching delays than systems which use a hot standby technique. The states labelled as D_1, D_2, ..., D_K are the states representing the switching delays. The total number of states is $4K - 2$.

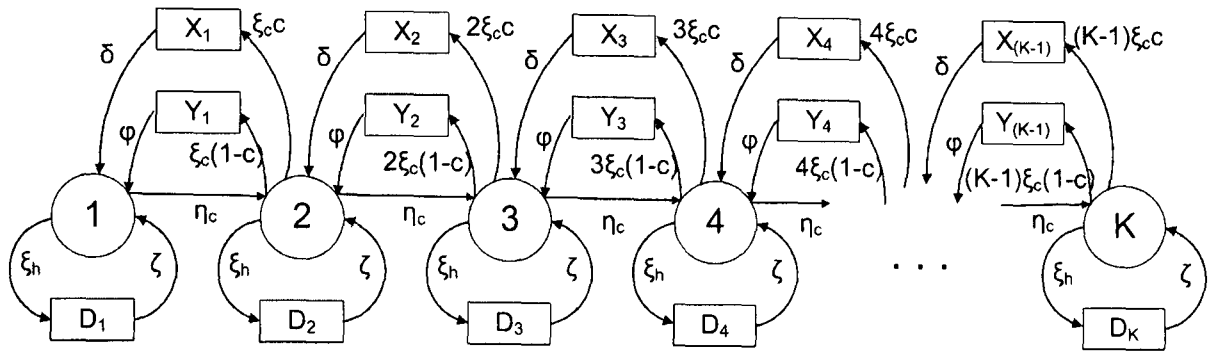

Figure 4.17: The operative states of a HA-Beowulf multiprocessor system with a serving head node

The model developed for the highly available Beowulf clusters with a non serving head node is very similar to the model shown in figure 4.17. However, since the head processor does not serve, there is no service rate assigned to state one. That means reconfiguration or rebooting is not necessary for transitions to state one, because the system does not serve at all. When there is one head and one identical node (i.e. state 2), in case of failure of the identical node the system will not continue serving in a degraded mode. The operative states of this model is given in figure 4.18. The total number of states is $4K - 4$.

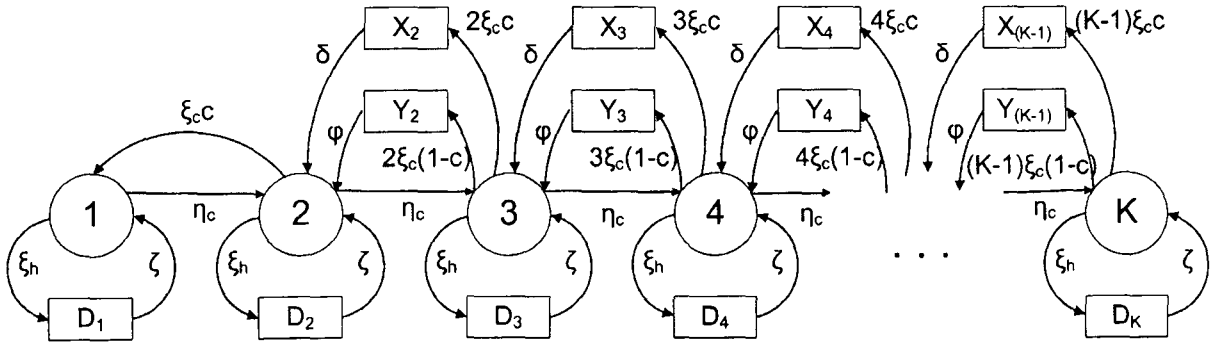

Figure 4.18: The operative states of a HA-Beowulf multiprocessor system with a non-serving head node

For highly available Beowulf systems with a head node which is not only allocating jobs but also serving jobs (i.e., system modelled in figure 4.17), we assign numbers (0, 1, 2, 3, ..., $K-1$) to the states 1, 2 , 3, ..., K respectively, and numbers (K, $K+1$, $K+2$, ..., $2K-1$) to the states D_1, D_2, ..., DK. The reconfiguration states X_1, X_2, ..., $X_{(K-1)}$ are represented by ($2K$, $2K+1$, ..., $3K-2$) and the rebooting states Y_1, Y_2, ..., $Y_{(K-1)}$ are given the numbers ($3K-1$, $3K$, ..., $4K-3$). This is a total of $4K-2$ states. For a highly available Beowulf cluster with a serving head node, the matrices B, C and A can be given as follows:

$B_j = B$ for $j = 0, 1, \ldots, L$; $B = \text{Diag}[\sigma, \sigma, \ldots, \sigma]$ of size $(4K-2) \times (4K-2)$.

$C_j = C$ for $j \geq K$; the threshold $M = K$, $C = \text{Diag}[\mu_h, (\mu_h + \mu_c), (\mu_h + 2\mu_c), \ldots, (\mu_h + (K-1)\mu_c), 0, \ldots, 0]$,

$C_0 = 0$.

If $\mu_h > \mu_c$, the head nodes have service priority over the identical ones. If the head node is idle, the incoming job goes to the head node and the remaining jobs receive service at the identical nodes. That gives rise to the following:

$C_j = \text{Diag}[(\mu_h + Min\{w_c(1), j-1\}\mu_c), (\mu_h + Min\{w_c(2), j-1\}\mu_c), \ldots, (\mu_h + Min\{w_c(K), j-1\}\mu_c), 0, \ldots, 0]$ for $1 \leq j < K$.

If $\mu_h \leq \mu_c$, then, the identical nodes have service priority over the head. In this case the incoming jobs first go to the identical nodes. If all of the identical nodes are busy, one of the jobs go to the head node.

$C_j = \text{Diag}[(Min\{w_c(1), j\}\mu_c + Min\{1, j - Min\{w_c(1), j\}\}\mu_h), (Min\{w_c(2), j\}\mu_c + Min\{1, j - Min\{w_c(2), j\}\}\mu_h), \ldots, (Min\{w_c(K), j\}\mu_c + Min\{1, j - Min\{w_c(K), j\}\}\mu_h), 0, \ldots, 0]$ for $1 \leq j < K$.

The number of identical processors available in the operative state i is $w_c(i)$. The last $3K-2$ diagonal elements of matrices C and C_j are always zero since they are used to represent the switching, reconfiguration and rebooting delay states. The matrix A is too bulky to be written in compact form, as an example, for $K = 4$, A can be written as follows:

$$A_j = A = \begin{pmatrix}
0 & \eta_c & 0 & 0 & \xi_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \eta_c & 0 & 0 & \xi_h & 0 & 0 & \xi_c c & 0 & 0 & \xi_c(1-c) & 0 & 0 \\
0 & 0 & 0 & \eta_c & 0 & 0 & \xi_h & 0 & 0 & 2\xi_c c & 0 & 0 & 2\xi_c(1-c) & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 & 0 & 3\xi_c c & 0 & 0 & 3\xi_c(1-c) \\
\zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

For highly available Beowulf systems with a head node which is only responsible for organisation and distribution of jobs, and does not provide service to incoming job requests (i.e., system modelled in figure 4.18), we assign numbers (0, 1, 2, 3, ..., $K-1$) to the states 1, 2, 3, ..., K respectively, and numbers (K,$K+1$, $K+2$, ..., $2K-1$) to the states (D_1, D_2 ,..., D_K). The reconfiguration states X_2 , ..., $X_{(K-1)}$ are represented by ($2K$, $2K+1$, ..., $3K-3$) and the rebooting states Y_2 , ..., $Y_{(K-1)}$ are given the numbers ($3K-2$, $3K-1$, ..., $4K-5$). This is a total of $4K-4$ states. This system is solved and the steady state probabilities, ($P_{i,j}$), are obtained using the Spectral Expansion solution. For a highly available Beowulf cluster with a serving head node, the matrices B, C and A can be given as follows:

$B_j = B$, $j = 0, 1, \ldots, L$; $B = \text{Diag}[\sigma, \sigma, \ldots, \sigma]$ of size $(4K-4) \times (4K-4)$.

$C_j = C$ for $j \geq K$; the threshold $M = K$

$C = \text{Diag}[0, \mu_c, 2\mu_c, \ldots, (K-1)\mu_c, 0, 0, \ldots, 0]$;

$C_0 = 0$

C_j =Diag$[(Min\{w_c(1), j\}\mu_c), (Min\{wc(2), j\}\mu_c), \ldots, (Min\{wc(K), j\}\mu_c), 0, \ldots, 0]$ for $1 \leq j < K$.

Here, $w_c(1)$ is zero since there are no computing nodes in state 1. In this system, since the main processor does not serve, there is no service at state 1. The last $3K-4$ diagonal elements of matrices C and C_j are always 0, since they are used to represent the switching and reconfiguration/rebooting delay states. Again, as an example, for $K = 4$, the matrix A can be written as follows:

$$A_j = A = \begin{pmatrix}
0 & \eta_c & 0 & 0 & \xi_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_c & 0 & \eta_c & 0 & 0 & \xi_h & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \eta_c & 0 & 0 & \xi_h & 0 & 2\xi_c c & 0 & 2\xi_c(1-c) & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 & 3\xi_c c & 0 & 3\xi_c(1-c) \\
\zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

4.3.2 Numerical Results and Discussions for Highly Available Beowulf Systems

In order to show the effectiveness of the models developed for highly available Beowulf clusters, to evaluate the performance of these systems, and to compare the performability measures of typical and highly available Beowulf clusters, numerical results are presented in this section. Similar to the computations for typical Beowulf systems, the parameters used are mainly taken for systems with highly reliable components. The models presented for the highly available systems also provide a large degree of flexibility and it is possible to obtain numerical results for systems with various characteristics. Highly available Beowulf clusters with unbounded queuing capacities are considered first.

Highly Available Beowulf Clusters with Unbounded Queuing Capacity

In figures 4.19 and 4.20, Beowulf clusters with 2, 3, 4 and 8-processors, are considered with non-computing and computing head nodes respectively. MQL values are computed for both highly available and typical Beowulf clusters for comparison purposes ((a) shows loaded systems, (b) shows systems with relatively lighter loads for figures 4.19, 4.20, 4.21, 4.22). Other parameters are given as $c = 0$, σ jobs/hour, $\xi_h = \xi_c = (1/6000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\mu_h = \mu_c = 300$ jobs/hr $\varphi = 12$/hr, and $\delta = 120$/hr. For highly available systems $1/\zeta$ is taken as 10 seconds (systems using hot standby technique are considered). To compare the performability of typical Beowulf clusters and highly available Beowulf clusters which use cold standby technique, figures 4.19 and 4.20 are regenerated with a switching delay $1/\zeta = 30$ minutes. Results are illustrated in figures 4.21 and 4.22.

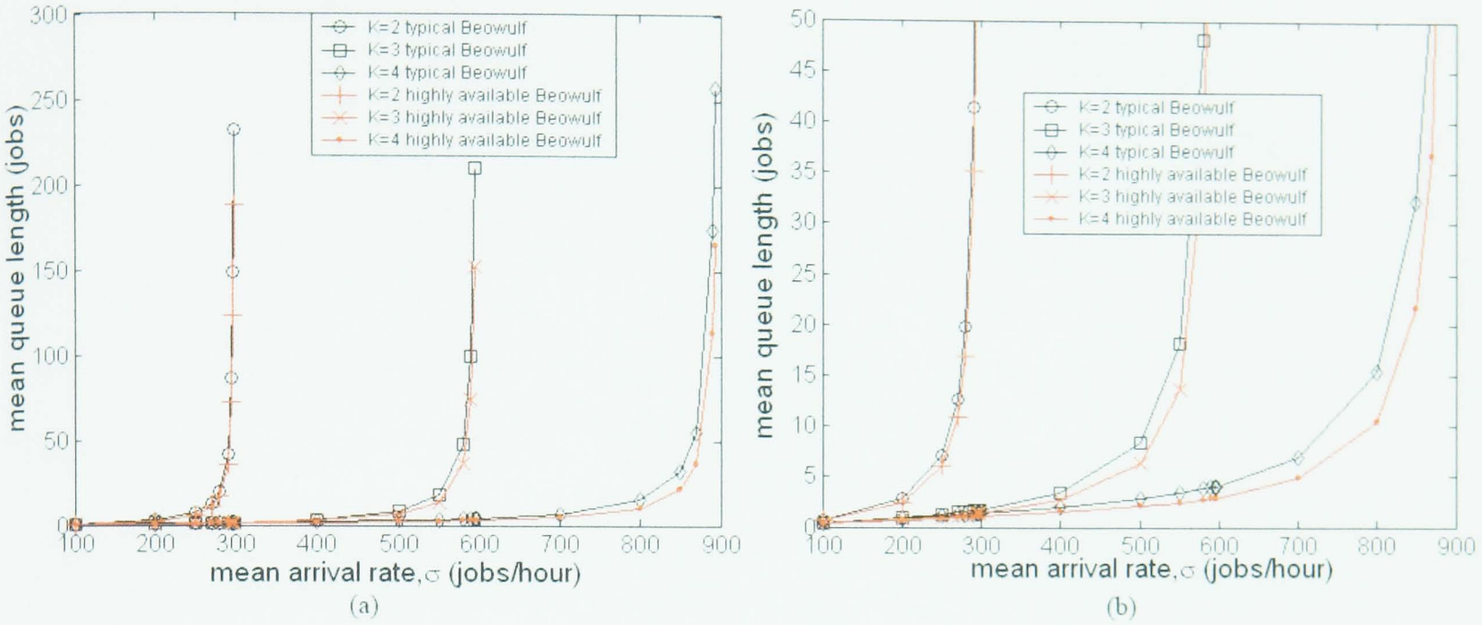

Figure 4.19: MQL versus mean arrival rate for both highly available (hot standby backup) and typical Beowulf systems with a non-serving head node.

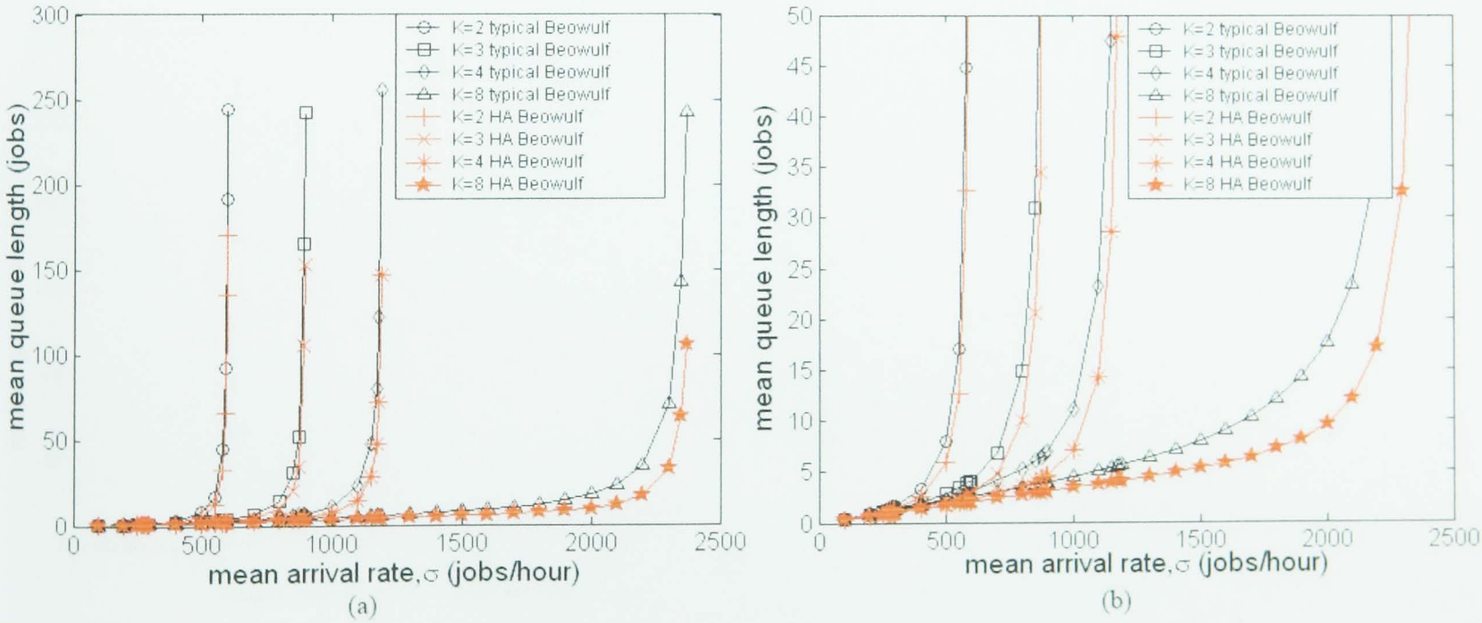

Figure 4.20: MQL versus mean arrival rate for both highly available (hot standby backup) and typical Beowulf systems with a serving head node.

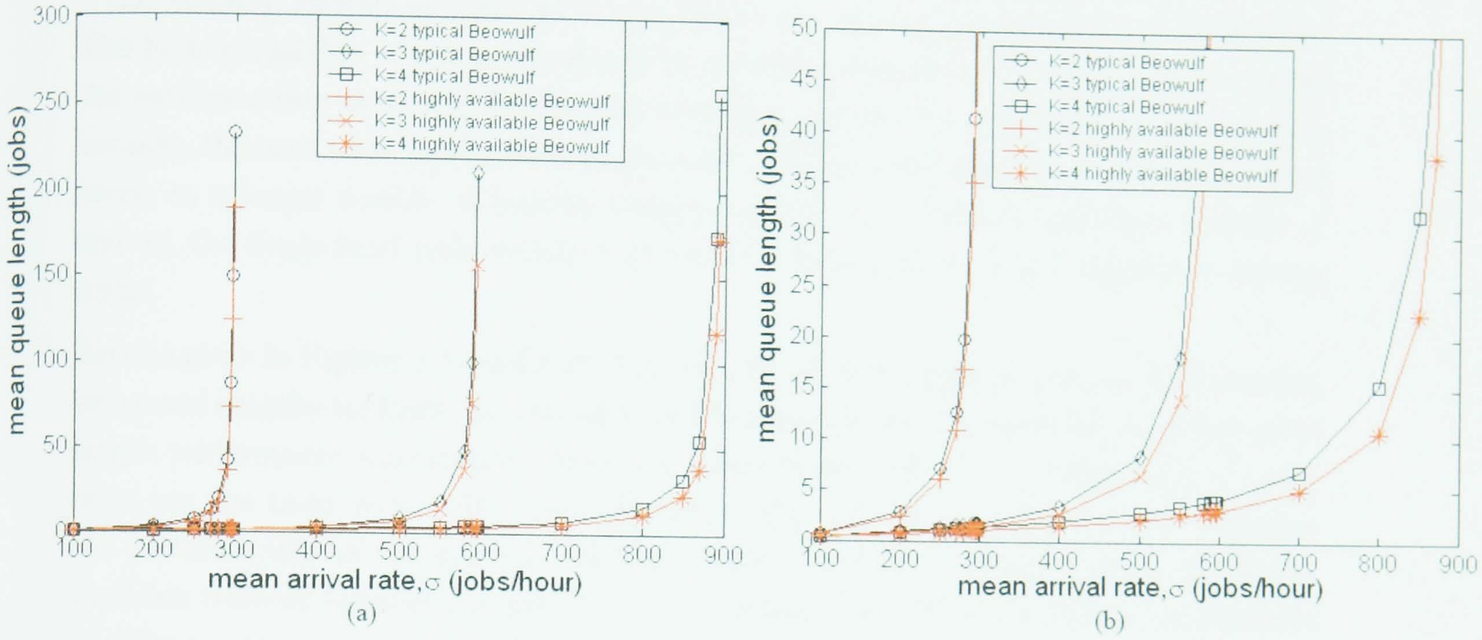

Figure 4.21: MQL versus mean arrival rate for both highly available (cold standby backup) and typical Beowulf systems with a non-serving head node.

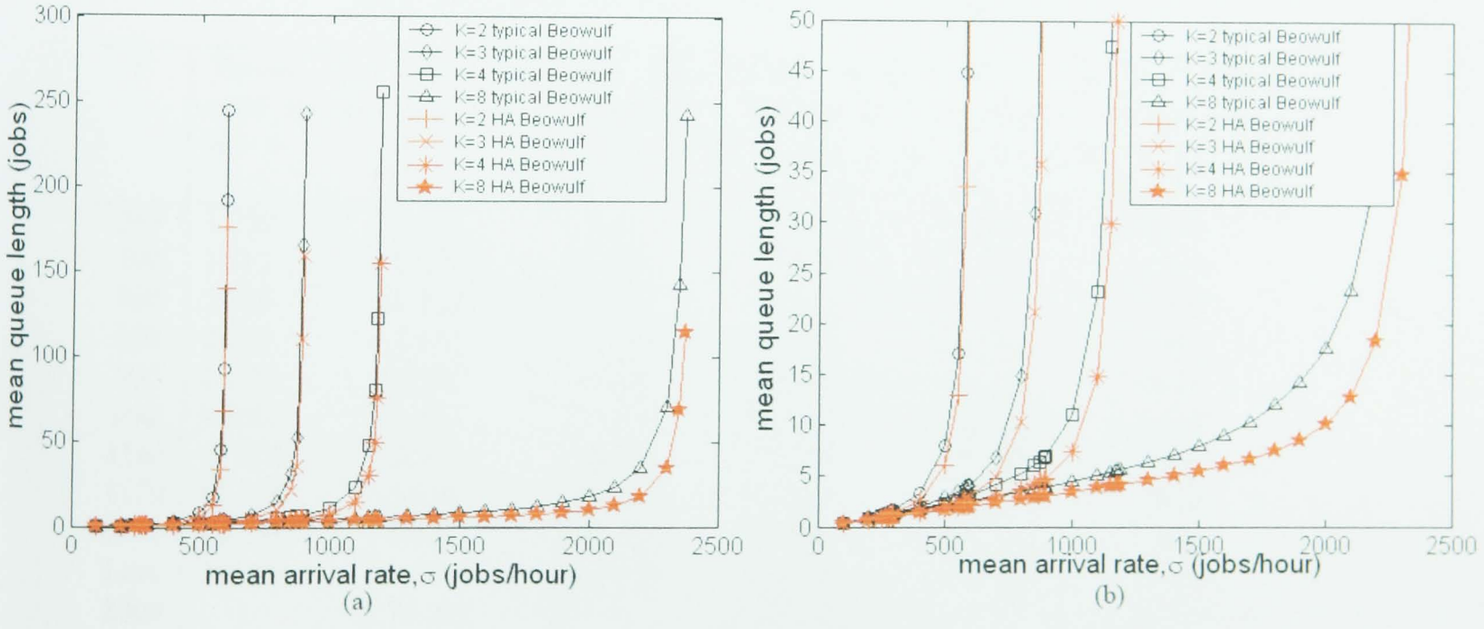

Figure 4.22: MQL versus mean arrival rate for both highly available (cold standby backup) and typical Beowulf systems with a serving head node.

Figures 4.19 and 4.20 show that in case of systems with unbounded queuing capacity, having a hot standby backup for head processor affects the systems performability significantly. The high availability which is introduced by providing redundancy to head node, improves the performability especially for systems with larger number of processors. This is mainly because, the head node represents a single point of failure and solving this problem provides access to a larger number of healthy identical nodes. For systems with larger number of servers, the single head node architecture becomes bottleneck for larger number of serving nodes.

Results given in Figures 4.21 and 4.22 show that although the replacement takes 30 minutes, using cold standby replacement technique for Beowulf systems, also improves the mean queue length performance significantly. Since the mean repair time $1/\eta_h$ is given as two hours, replacing the head node with mean switching delay $1/\zeta$ which is half an hour improves the performability of the systems. Table 4.2 shows the MQL values of typical and highly available Beowulf systems in order to further emphasize the differences in the performance of different systems. In case of HA Beowulf Systems, switching delays $(1/\zeta)$ of 10 seconds and 30 minutes have been considered for hot and cold standby systems respectively. For all calculations, $c = 0$, σ jobs/hour, $\xi_h = \xi_c = (1/6000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\mu_h = \mu_c = 300$ jobs/hr $\varphi = 12$/hr, and $\delta = 120$/hr have been assumed.

Table 4.2: MQL values for various Beowulf systems

σ	*Typical Beowulf System* $K = 4$	*Typical Beowulf System* $K = 8$	*HA Beowulf System* $K = 4$, $1/\zeta = 10$ *sec*	*HA Beowulf System* $K = 8$, $1/\zeta = 10$ *sec*	*HA Beowulf System* $K = 4$, $1/\zeta = 30$ *min*	*HA Beowulf System* $K = 8$, $1/\zeta = 30$ *min*
290	1.229	1.189	0.974	0.969	0.989	0.983
400	1.762	1.657	1.361	1.337	1.386	1.357
500	2.316	2.093	1.744	1.672	1.779	1.698
590	2.908	2.495	2.133	1.973	2.182	2.00
800	5.229	3.480	3.620	2.680	3.721	2.731
890	6.825	3.93	4.523	2.986	4.668	3.045
1150	47.508	5.371	28.615	3.898	29.83	3.990
1170	80.476	5.494	47.855	3.971	49.99	4.066
1180	122.637	5.557	72.151	4.008	75.528	4.104
1190	255.94	5.62	146.79	4.045	154.531	4.143
2300		70.221		32.556		34.939
2350		142.417		63.565		68.615
2370		242.764		105.795		114.703

Results clearly illustrate the superior performance of HA Beowulf Systems using hot standby replacement approach. The low breakdown rates chosen have resulted in $5 - 8\%$ difference between hot and cold standby systems, this figure is expected to increase for less reliable systems showing the importance of hot standby systems. When HA systems are compared to typical Beowulf systems, the performance improvement provided by the former is far superior, at the cost of an additional, standby server.

To show the effects of the switching delay $1/\zeta$ on systems performance, eight node systems are considered. Mean queue length values are computed as a function of switching delay for systems with a serving head node and different arrival rates. Other parameters are given as $c = 1$, σ jobs/sec, $\xi_h = \xi_c = (1/1000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\mu_h = \mu_c = 8000$ jobs/hr $\varphi = 12$/hr, and $\delta = 120$/hr.

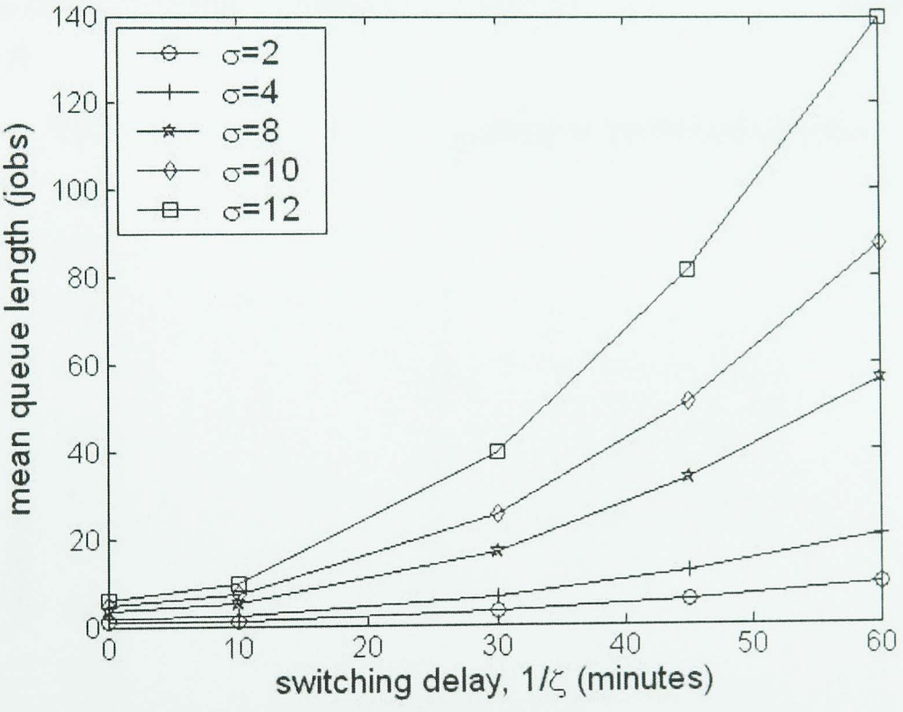

Figure 4.23: MQL as a function of $1/\zeta$ and σ for highly available Beowulf systems with a serving head node and $K = 8$.

Figure 4.23 shows how the effects of switching delay increases as the mean arrival rate increases. While networks with a light load can afford longer switching delays, for loaded networks, standby replacement approach used becomes more critical for system performance.

The overall results show the importance of the use of highly available Beowulf systems for systems with unbounded queuing capacities, even if cold standby technique is used. This is more significant for systems with high arrival rates and larger number of processors. Also for loaded systems the effects of switching delay (hot, cold standby approaches) are relatively higher.

Highly Available Beowulf Clusters with Finite Queuing Capacity

Beowulf multi-server systems with bounded queues are considered. Numerical results are presented for MQL and PJL measures.

In figures 4.24 and 4.25, the mean queue length is calculated for various σ values with $K = 3$ and 4, $\mu_h = \mu_c = 300$ jobs/hr, $\xi_h = \xi_c = (1/1000)$/hr, $\eta_h = \eta_c = 0.5$/hr, $\varphi = 3$/hr, $\delta = 120$/hr, $c = 1$ and $L = 1000$. Highly available Beowulf systems are considered as well as typical Beowulf clusters. In figures 4.26 and 4.27 PJL results are presented for the same parameters.

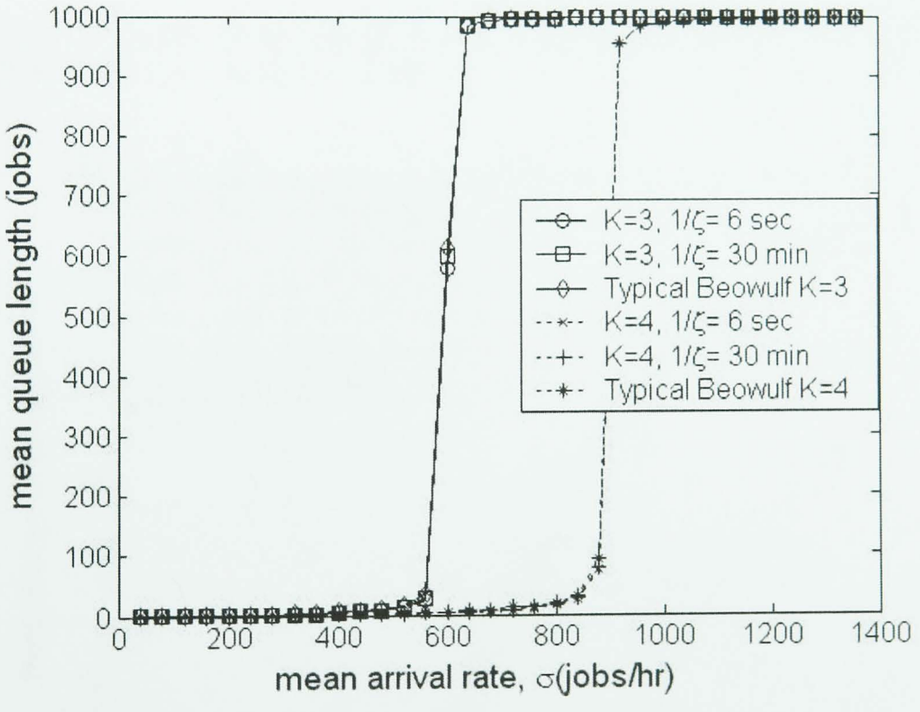

Figure 4.24: MQL as a function of K and σ for typical and HA Beowulf systems with a non-serving head node $c = 1$ and $L = 1000$.

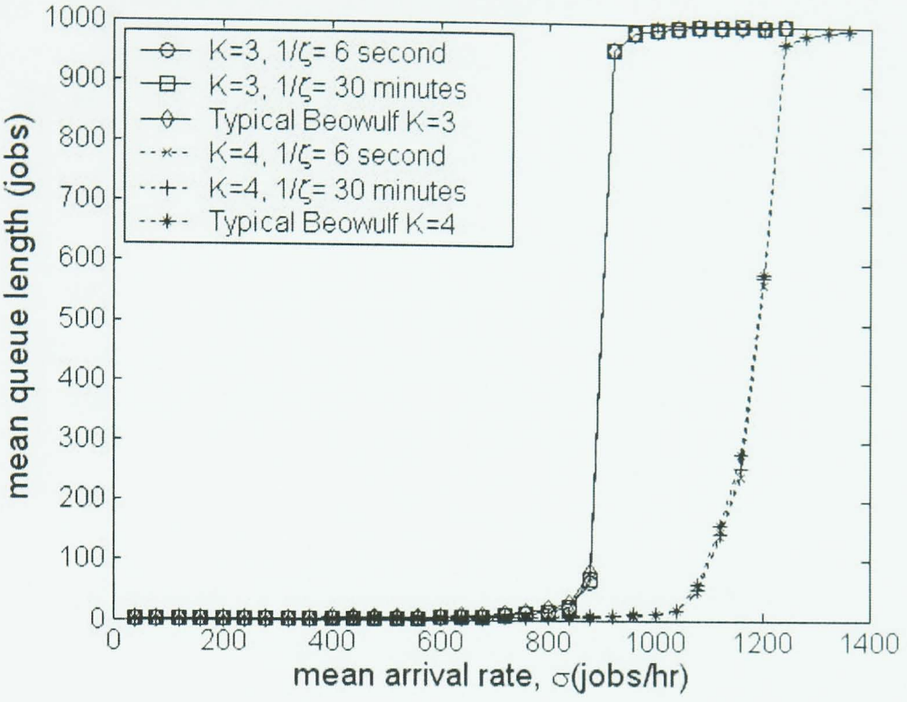

Figure 4.25: MQL as a function of K and σ for typical and HA Beowulf systems with a serving head node $c = 1$ and $L = 1000$.

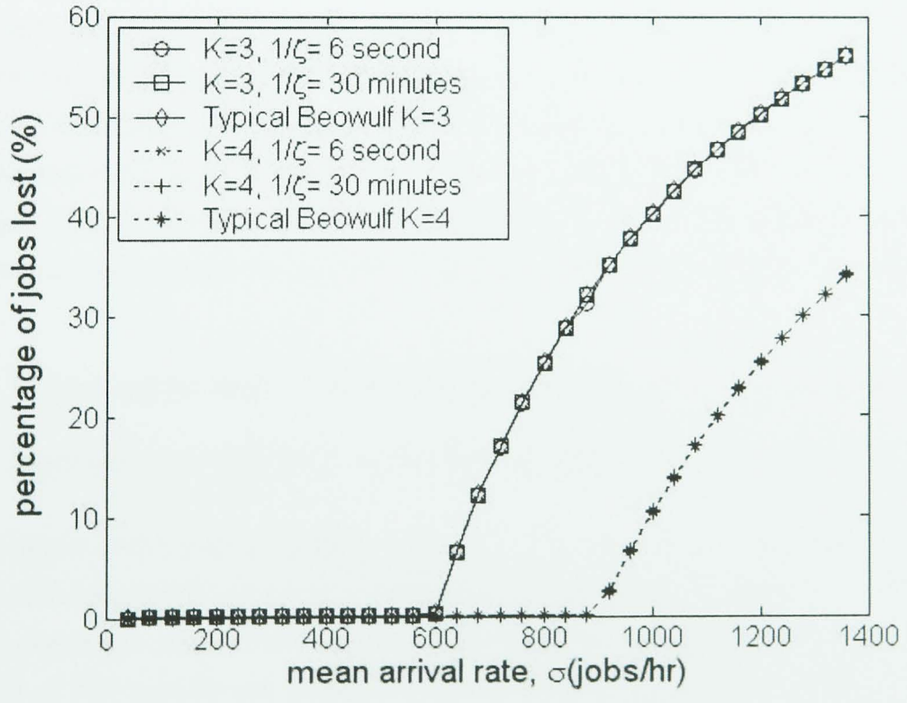

Figure 4.26: PJL as a function of K and σ for typical and HA Beowulf systems with a non-serving head node $c = 1$ and $L = 1000$.

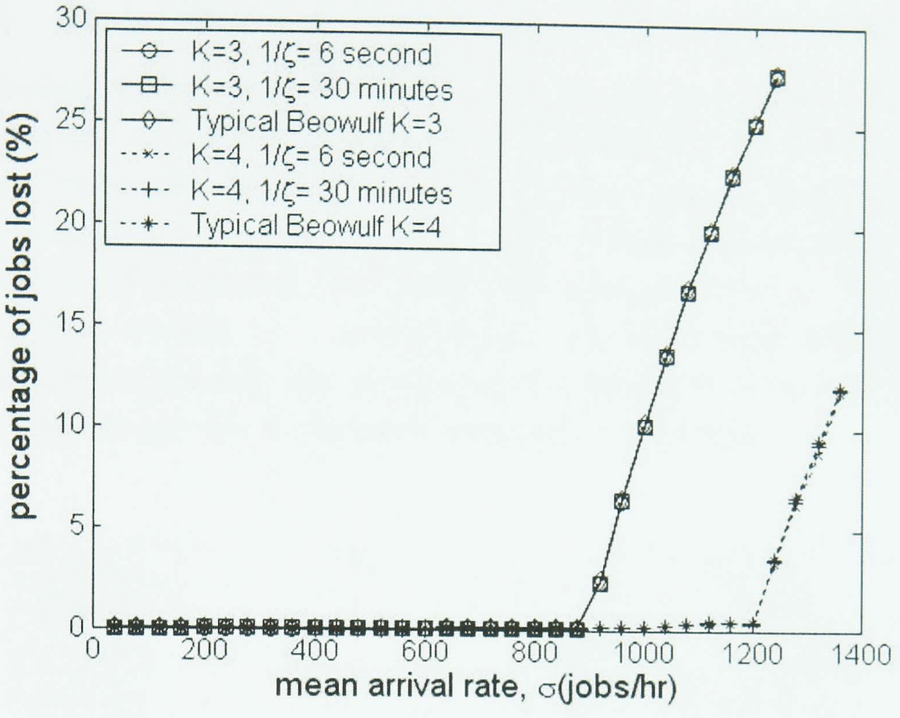

Figure 4.27: PJL as a function of K and σ for typical and HA Beowulf systems with a serving head node $c = 1$ and $L = 1000$.

The results in figures 4.24 and 4.25 show that having hot or cold standby facilities for the head server do not necessarily improve the mean queue length performance in case of systems with bounded queuing capacities. The queuing capacity L is the main limiting factor for highly available Beowulf clusters as well for loaded networks. For relatively small arrival rates, systems with greater number of processors perform better, but in case of high arrival rates again L becomes the main limiting factor. Figures 4.26 and 4.27 confirm these results presenting the percentage of jobs lost in highly available and typical Beowulf cluster systems.

4.4 Discussions and Conclusions on Farm Paradigm Systems with One Head and Several Identical Servers

In this chapter analytical models are presented for typical and highly available multi-server systems with one head and several identical servers. Beowulf clusters are taken as case study for farm paradigm multi-server systems. Scalability of the presented models discussed and validation of the models are presented in the following section.

4.4.1 Scalability and Validation of Proposed Models

The method can be extended to calculate the performance of larger Beowulf systems when rebooting and reconfiguration delays are not considered. Figure 4.28 shows the MQL performance of typical Beowulf clusters with 12, 16, 20, and 24 processors including a serving head. In this figure, simulation results are also presented comparatively in order to validate the accuracy of the models considered. Other parameters are, $\xi_c = 0.001$/hr, ξ_h/hr is variable, $\eta_c = \eta_h = 0.5$/hr, $\mu_c = \mu_h = 5$ jobs/hr. The effect of the reliability of the head node is illustrated on the diagram. As expected the reliability of the head processor affects the system significantly due to the single head node architecture.

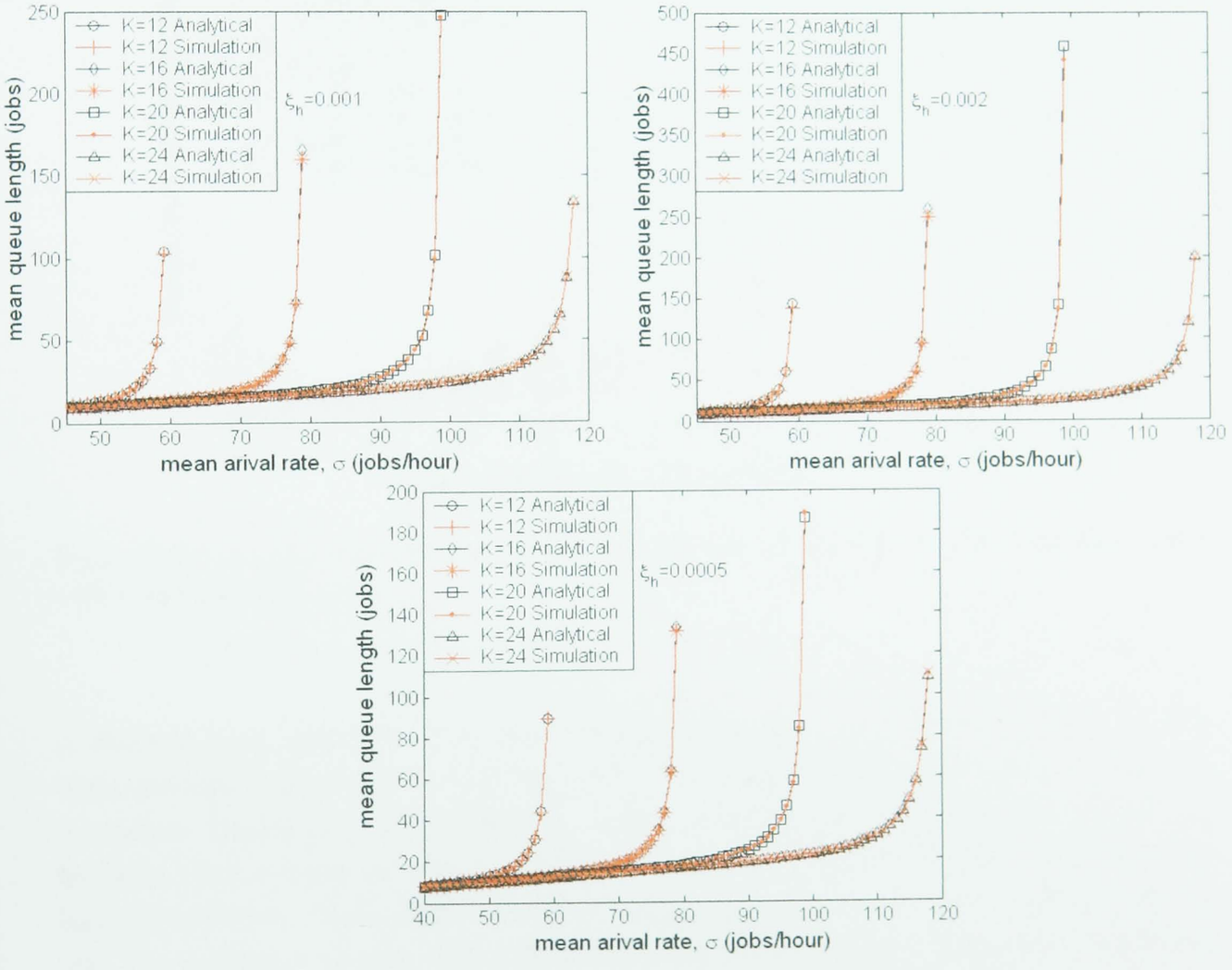

Figure 4.28: MQL performance of larger scale Beowulf clusters.

Finally, a comparative study is preformed in order to validate the proposed technique and evaluate the accuracy of the results obtained for the systems with reconfiguration and rebooting delays. Simulation and analytical results are presented comparatively for typical Beowulf clusters with a serving head node, in figure 4.29. The mean queue length is calculated as a function of σ (jobs/hr) for systems with various failure rates ($\xi = \xi_h = \xi_c$/hr). Other parameters are $\mu_h = \mu_c = 5$ jobs/hr, $\eta_h = \eta_c = 0.5/hr$, $\varphi = 12$/hr, $\delta = 120$/hr, $c = 1$.

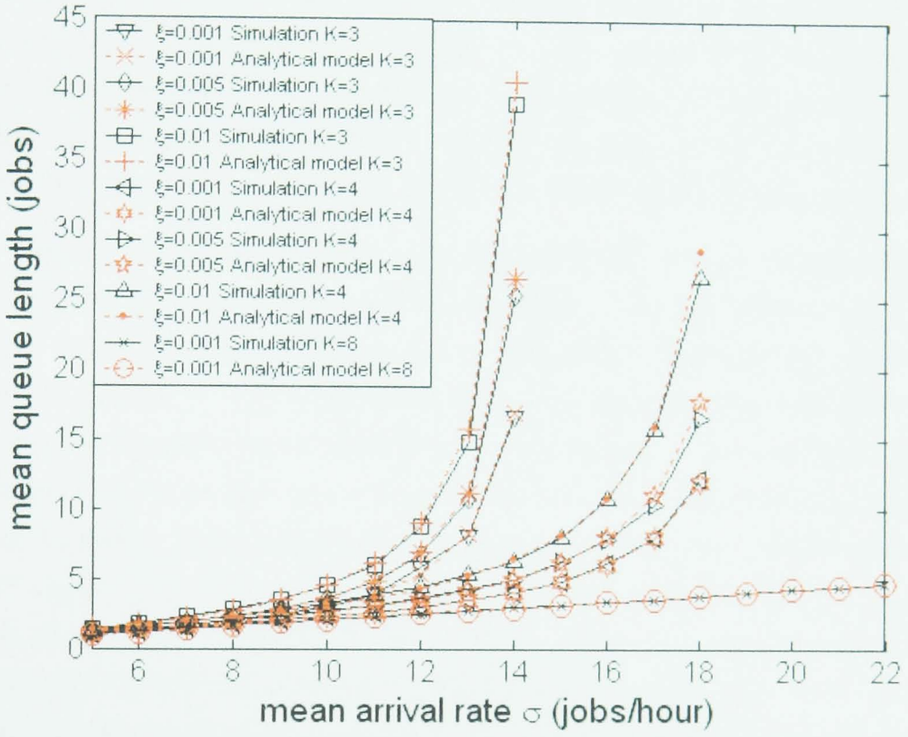

Figure 4.29: Results from simulation and mathematical model for typical Beowulf clusters with a serving head node.

A discrete event simulation programme implementing an event scheduling algorithm has been specifically developed for the validation of the analytical models presented. First, the simulation programme was validated by means of the use of well known queuing theory formulae (Deitel 1990) as well as results from published material (Chakka 1995, Chakka and Mitrani 1996, Chakka 1998). The comparisons showed that these show good agreement with the simulation results. The validated simulations are used for validation of proposed analytical models presented throughout this thesis.

Results from the proposed analytical approach and simulation show good agreement. For example, for $\sigma = 16$ jobs/hour and 160 replications, 95% confidence level has been achieved

for a confidence interval of (10.667, 10.679). For simulations, the scenario explained in previous sections is considered. The simulation results are given for the actual process and not for the Markov model used for analytical studies, and they are within the confidence interval of $\pm 5\%$ of the mean value with a confidence level of 95%.

4.4.2 Conclusions and Recommendations

Typical and highly available farm paradigm systems with one head and several identical processors have been studied in this chapter. A new approach is presented for analytical modelling of these systems. The steady state probabilities for typical and highly available farm paradigm systems with breakdowns, repairs, reconfiguration and rebooting delays are derived using the QBD processes coupled with the Spectral Expansion method.

Numerical results have been obtained and presented for various performability measures, for both bounded and unbounded queuing capacities. Results show that, for typical Beowulf clusters, when queue capacity is not an important factor on the mean queue length performance, the choice of the optimum number of processors depends on the values of reconfiguration and rebooting delays as well as c, and to have a serving head node can have a significant impact on the system performance even if the number of serving processors is kept same in total. However, for bounded queuing systems, L is the main factor affecting the mean queue length performance of the systems especially for relatively large arrival rates. Also results obtained for highly available Beowulf clusters show that having backup processors for the head node can improve the systems' performability significantly. Systems with hot and cold standby processors are also compared. In case of non blocking systems, multi-server systems with hot standby backup head node performs significantly better than systems with cold standby head nodes. However, for systems with bounded queuing capacity, again, L is the main factor affecting the performability measures for highly available Beowulf clusters. Having standby processors cannot improve the system performance especially if high arrival rates are expected. The models developed are highly flexible and they can be used for performability evaluation of typical and highly available Beowulf clusters with various characteristics such as systems with serving or non-serving head nodes, systems with components which have various reliabilities and various service rates. These numerical results and the methodology can be of great help to modify/improvise the operating system and operational aspects effectively, according to the performability and availability requirements.

The method is useful for the designers and users of small scale Beowulf systems which are frequently used by scientific and computing communities. The method can be further extended to many of the high performance/highly available/highly reliable computer architectures,

with appropriate modifications. The same approach can be used in modelling various kinds of cluster systems. An important limitation of the technique is scalability which is restricted because of state space explosion problem. However, this problem is common for models representing 2-dimensional Markov processes. Approximate solutions are available to ease this problem. One such approach is presented in turn.

Chapter 5

Modelling Open Queuing Networks with Unreliable Servers and Finite Buffers

Networks of queues are extensively used in modelling transaction processing systems, and the interactions among nodes in communication networks. Many practical networks, such as networks with finite capacity, or, with general arrival and service processes, do not fall into the category of product form networks. Also when the servers in a network are prone to breakdowns and repairs, the irregularities caused by server breakdowns and repairs have a great effect on the performance and dependability of the network, and such a network does not have a product form solution. The analysis and solution of such networks is important since many practical systems fall into this category.

In such a system at each node, it is possible to approximate the traffic as the superposition of external arrivals and internal arrivals fed by other nodes, by considering it as a single stream with bursty nature. In (Ny, and Sericola 2002, Dudin et. al. 2005, Klimenok et. al. 2005) various approaches are given to model systems with bursty arrivals. However in these approaches, server breakdowns and repairs are not considered.

Open network systems with breakdowns and repairs are effectively used to model various kinds of practical computer and communication networks (Chakka 1995, Kelly 1996). Some approximate analytical solutions, supported by validation through simulation, are suggested for these cases. Open networks with infinite queuing capacities are considered and approximate solutions are presented in (Chakka 1995, Chakka and Mitrani 1996). Two approaches are given for open network systems where each node has an unbounded queue. The first approach is a performance model based on the use of MMPP to represent various bursty

arrival and departure processes. The second approach is called joint state approach. In this approach, a binary random variable which is equal to zero in case the server is broken and one in case the server is operative is used, to represent the operative states of the servers at each node (Chakka 1995). In this chapter these studies are extended for open networks with finite queuing capacities. An approximate model and solution for the steady state probabilities of open network systems with breakdowns and repairs, and with bounded queuing capacities are driven. Previous studies are extended and an efficient iterative procedure that employs the Spectral Expansion solution algorithm as a major computation tool, is developed. For both IPP and joint state approaches, threshold value M is 1 since each node has only one server.

Since the Markov models of many practical systems have finite buffer space, rather than an infinite one, in this study several other engineering techniques are used together with Spectral Expansion method in order to find an approximate solution for the performability analysis of open networks with bounded buffers, breakdowns and repairs. Numerical performability results are presented for such systems. Simulation results have been obtained to establish a certain degree of accuracy for the models considered.

5.1 Open Queuing Network System Under Study

In this study, the system analysed is an open network with K nodes where each node contains a server and a bounded FIFO queue. In such a system the total flow of jobs is the superposition of external (Poisson) and internal flows from other nodes. In many practical systems, because of the feedbacks from the other nodes, the arrival rates to a node may vary randomly over time quite sharply sometimes. Examples include superposition of packet oriented voice processes, and packet data in communication modelling (Chakka and Mitrani 1996).

The servers are numbered as, 1, 2, ..., K. Numbering can be arbitrary or with respect to any possible convenience. The service, breakdown and repair times of the servers at node k are exponentially distributed with rates μ_k, ξ_k, and η_k, respectively. The queuing capacity of the k^{th} node is finite, given by L_k. Each server serves one job at a time when it is operative. The external arrival rate to node k is σ_k. When a job's service is interrupted because of a server breakdown, its service is resumed when a server is available, on the basis of resume or repeat with re-sampling discipline. After a job is serviced at node k, it is routed to node l with a probability of $q_{k,l}$. The probability of a job to leave the system after being served at node k is $q_{k,K+1} = 1 - \sum_{l=1}^{K} q_{k,l}$. A square matrix Q of size $K \times (K+1)$ is formed as,

$Q(k,l) = q_{k,l}$ $(k,l = 1,2,\ldots,K)$ where $(K+1)^{th}$ column represents the departures which are not fed to the nodes in the system. Q is the routing probability matrix strictly among the nodes. This system is shown in Figure 5.1 , for $K = 3$.

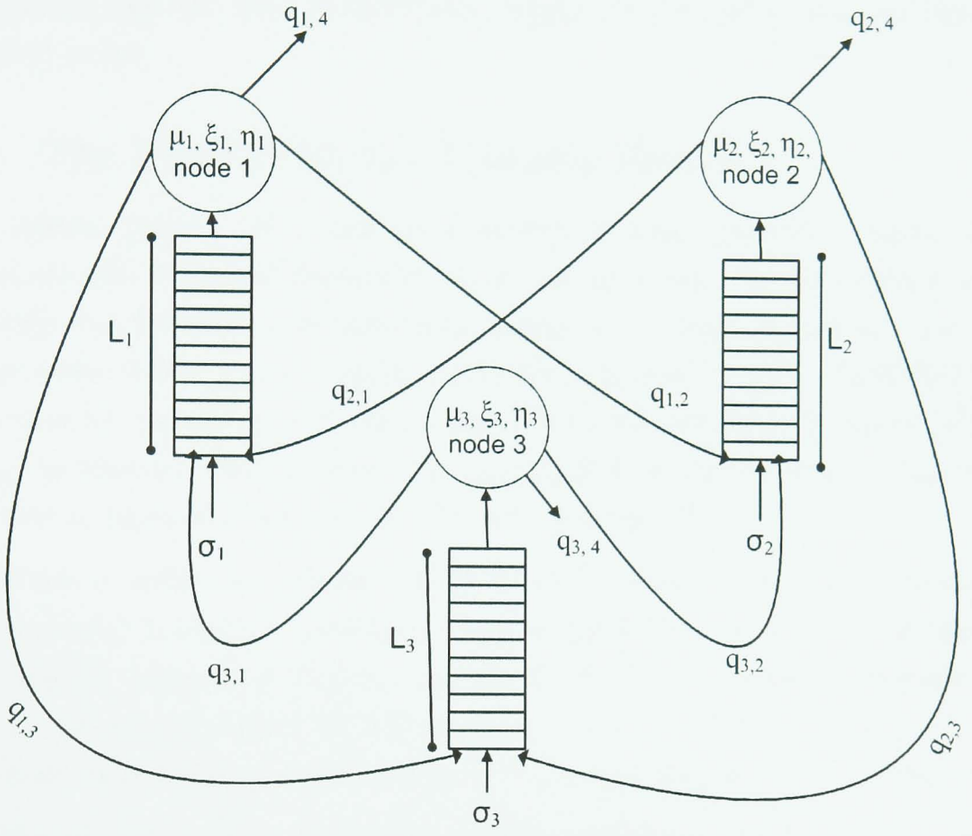

Figure 5.1: Open queuing network considered

5.2 Modelling Open Networks with Breakdowns, Repairs and Finite Buffers, Using an IPP Departure Process Model

In this approach, an MMPP is used to represent the traffic into a node consisting of Poisson external arrivals together with feedbacks expected from other nodes.

Each node in the system is then treated as an $MMPP/M/1/L$ queuing model with breakdowns and repairs. When a server breaks down, it is set into repair process immediately.

After the completion of the repair, the server becomes operative again. The Interrupted Poisson process (IPP) is used for departure process modelling of these nodes. This approach has been used for performability evaluation of open queuing systems with unbounded queuing capacities in (Chakka and Mitrani 1996, Chakka et. al. 2000), with some success. However open queuing systems with breakdowns, repairs and bounded queuing capacities are not considered so far.

5.2.1 The MMPP/M/1/L Queuing System

In the system under study, each node receives Poisson external arrivals. In addition to external arrivals, feedbacks from other nodes are expected. The jobs receive service with exponentially distributed service times and, if they do not leave the system, they are forwarded to other nodes following the routing probability matrix Q. Since $MMPP/M/1/L$ queues can be used for modelling such nodes, first a bounded single server queue, with MMPP job arrivals has been analysed, to show the limiting effect of having bounded queuing capacities. The server considered is prone to breakdowns and repairs.

The number of states (or, phases) of the MMPP arrival process is represented by T. The $MMPP$ process is characterised by the square matrices Θ and Σ. Θ is the generator matrix of the MMPP, which is a diagonal matrix of size $T \times T$. The i^{th} diagonal element of Σ represents the arrival rate of MMPP in phase i. If c_i, $i = 0, 1,2, \ldots, T-1$ is defined as the steady state probability that the system is in phase i, then the total average arrival rate can be computed as $\hat{\sigma} = \sum_{i=0}^{T-1} c_i \sigma_i$ (Fischer and Meier-Hellstern 1993, Chakka 1995, Chakka et. al. 2000).

The state of this system can be specified completely by three discrete random variables (Chakka and Mitrani 1996).

i. The phase of the arrival process, $\Phi(t)$ which is an integer random variable varying from 0 to $T-1$.

ii. A boolean variable, $O(t)$, which shows whether the server is broken or operative.

iii. The number of jobs in the system including the one being served, $J(t)$ ($J(t)$ in $(0, 1, 2, \ldots, L)$) where L is the queuing capacity.

Random variables $\Phi(t)$and $O(t)$ can be combined since they are of finite range. The first T states of $I(t)$ are used to represent the breakdown states of the server, and remaining T states are used to represent the operative states with respective arrival phases of the MMPP. This

is a QBD process where ($I(t) = 0, 1, \ldots, 2T$, $J(t) = 0, 1, \ldots, L$). The Spectral Expansion method is used to solve this system for state probabilities. The transition matrices B, C, and A are defined as follows:

$B_j = B$ for $j = 0, 1, \ldots, L$;

$B = Diag[\sigma_0, \sigma_1, \ldots, \sigma_{T-1}, \sigma_0, \sigma_1, \ldots, \sigma_{T-1}]$ of size $2T \times 2T$.

$C = Diag[0, 0, \ldots, 0, \mu, \mu, \ldots, \mu]$ of size $2T \times 2T$, where the first T elements of matrix C are zero since they represent the states with broken server.

$C_j = C$ for $j \geq 0$, $C_0 = 0$.

$$A_j = A = \begin{pmatrix} \Theta_{0,0} & \Theta_{0,1} & \cdots & \Theta_{0,T-1} & \eta & 0 & 0 & 0 \\ \Theta_{1,0} & \Theta_{1,1} & \cdots & \Theta_{1,T-1} & 0 & \eta & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & 0 & \ddots & 0 \\ \Theta_{T-1,0} & \Theta_{T-1,1} & \cdots & \Theta_{T-1,T-1} & 0 & 0 & 0 & \eta \\ \xi & 0 & 0 & 0 & \Theta_{0,0} & \Theta_{0,1} & \cdots & \Theta_{0,T-1} \\ 0 & \xi & 0 & 0 & \Theta_{1,0} & \Theta_{1,1} & \cdots & \Theta_{1,T-1} \\ 0 & 0 & \ddots & 0 & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \xi & \Theta_{T-1,0} & \Theta_{T-1,1} & \cdots & \Theta_{T-1,T-1} \end{pmatrix}$$

An example can be considered in order to compare $MMPP/M/1/L$ systems with various queuing capacities. The system considered has a 3-phase MMPP arrival process. The phases are numbered as, 0, 1, and 2. Phase 1 is the normal phase of the arrival process, with arrival rate σ. In phase 2, the arrivals are high, 10σ. In phase 0, the arrival rate is low given as, 0.1σ. The transition rates are given as;

$$A_j = A = \begin{pmatrix} 0 & 0.1\Theta & 0.1\Theta \\ 0.007\Theta & 0 & 0.003\Theta \\ 0.05\Theta & 0.05\Theta & 0 \end{pmatrix}$$

In Figure 5.2 the mean queue length is computed for various values. Other parameters are $\mu = 25$ jobs/hr, $\xi = 0.01$/hr, $\eta = 0.1$/hr, $\Theta = 1.0$. $MMPP/M/1/L$ models and $M/M/1/L$ models have been considered with various queuing capacities.

These results show that the $M/M/1/L$ results are gross underestimation of the $MMPP/M/1/L$ queue. Clearly, the burstiness of the arrivals (in the case of open networks, this is caused by feedbacks from other nodes) affects the performance measures considerably. It also shows that in case of bounded systems the size of the queue is the main limiting factor. However, even in case of finite queues MQL measures of $MMPP/M/1/L$ are relatively higher than MQL measures of $M/M/1/L$. This shows that even in case of finite queuing capacities, having bursty arrivals caused by feedbacks, will affect the system's performance significantly.

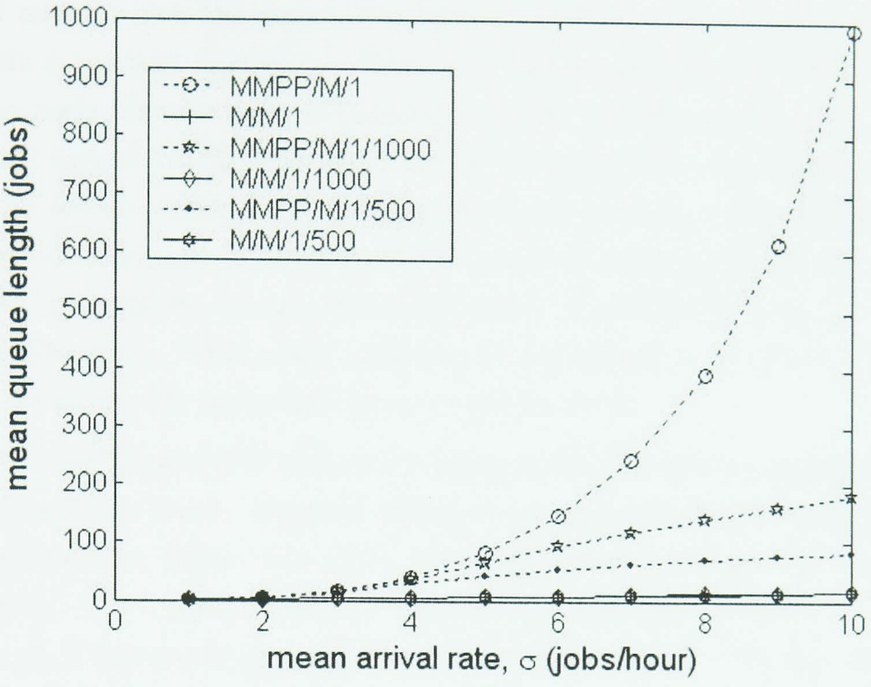

Figure 5.2: MQL for MMPP/M/1/L and M/M/1/L

5.2.2 The Departure Process Modelling and Solution Using IPP

In the system considered (shown in figure 5.1) there are two kinds of arrivals expected at each node. Arrivals can be from outside the system or there may be feedbacks from other nodes in the system. Since the departed job can be arrival process for other nodes, departure process modelling is an important issue for analysing open networks. In this section the IPP is chosen as the model for the departure process which can be defined as a two-phase MMPP where flow rate in one of the phases is equal to zero. The parameters of this departure process are taken as υ, α, and β. In phase 1 the departure rate is υ. In phase 2 the departure rate is zero. Transition rate from phase 1 to phase 2 is α, and from phase 2 to phase 1 is β. The details about the computations of these parameters are given in section 2.3.4.

The iterative procedure given in (Chakka 1995, Chakka and Mitrani 1996, Chakka et. al. 2000) can be used to solve the model representing open queuing networks with breakdowns, repairs and finite queuing capacity as well. Once the steady state probabilities are computed, various performance evaluation measures can be obtained. For this study independent Poisson processes for external arrivals to the nodes are used. The departures from the nodes are approximated to an IPP following the IPP model 1 in (Chakka and Mitrani 1996). To be able to use IPP model for a departure process, some assumptions are made. Let S_k be

the set of nodes which the departures from node k are routed and G_k be the set of nodes which node k receives jobs from. Then, the split process of the IPP departure from node k, entering node l is also approximated by an IPP with parameters $\upsilon_{k,l}$, $\alpha_{k,l}$ and $\beta_{k,l}$. Also, the internal arrival $IPP_{k,l}$ and the external Poisson arrival to node k are assumed to be independent for each node k. With these assumptions, node k receives one independent IPP from each node belonging to set G_k plus its independent external Poisson. The superposition of all these independent streams entering node k, is an MMPP with 2^{g_k} phases, where g_k is the cardinality of G_k. Then, each node can be considered as an $MMPP/M/1/L$ model and solved by following the procedure given in section 2.3.4.

Probabilistic splitting of IPP and constructing MMPP are very important issues used in the iterative procedure given. Detailed information about these processes is given in (Fischer and Meier-Hellstern 1993). Let IPP_k be the departure process of node k with parameters υ_k, α_k and β_k. Then, parameters of $IPP_{k,l}$ are $\upsilon_{k,l}$, $\alpha_{k,l}$ and $\beta_{k,l}$ where $\upsilon_{k,l} = \upsilon_k\ q_{k,l}$, $\alpha_{k,l} = \alpha_k$ and $\beta_{k,l} = \beta_k$ (Fischer and Meier-Hellstern 1993). Let $MMPP_k$ be the overall arrival process of node k, the arrival rate in phase n (n = 0, 1, ..., $T_k - 1$) be $\sigma_{k,n}$, Σ_k be the diagonal matrix, of size $2^{g_k} \times 2^{g_k}$, whose n^{th} diagonal element is $\sigma_{k,n}$, and Θ_k be the generator matrix, of size $2^{g_k} \times 2^{g_k}$, of $MMPP_k$. Then, the matrices Θ_k and Σ_k can be expressed as:

$$\Sigma_k = \Sigma_{(1,k)} \bigoplus \Sigma_{(2,k)} \bigoplus \ldots \bigoplus \Sigma_{(g_k,k)} + \sigma_k I_k,\ \Theta_k = \Theta_{(1,k)} \bigoplus \Theta_{(2,k)} \bigoplus \ldots \bigoplus \Theta_{(g_k,k)}$$

where, $\bigoplus$ stands for Kronecker sum, σ_k is the arrival rate of external arrivals to node k, and I_k is the unit matrix with the same size as Σ_k. With the above procedures to compute $MMPP_k$ and $IPP_{k,l}$, it is possible to use the iterative solution given in 2.3.4 to obtain the mean queue length measures of each node which receives MMPP arrivals.

5.2.3 Numerical Results for Open Networks with IPP Departures and Finite Buffers

In this section numerical results are presented for three server tandem networks with breakdowns and repairs. The PJL values are computed for each node. The accuracy of IPP departure model used for open networks with finite queuing capacities is tested in following sections together with joint state approach. In all computations the tolerance value is taken as 0.01. Service rates, failure rates and repair rates of nodes 1, 2, and 3 are $\mu_1 = 5.0$ jobs/hr, $\xi_1 = 0.001$/hr, $\eta_1 = 0.002$/hr, $\mu_2 = 6.0$ jobs/hr, $\xi_2 = 0.002$/hr, $\eta_2 = 0.003$/hr, $\mu_3 = 7.0$ jobs/hr, $\xi_3 = 0.003$/hr, $\eta_3 = 0.005$/hr respectively. The routing probability matrix Q is:

$$Q = \begin{pmatrix} 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 1.0 \end{pmatrix}$$

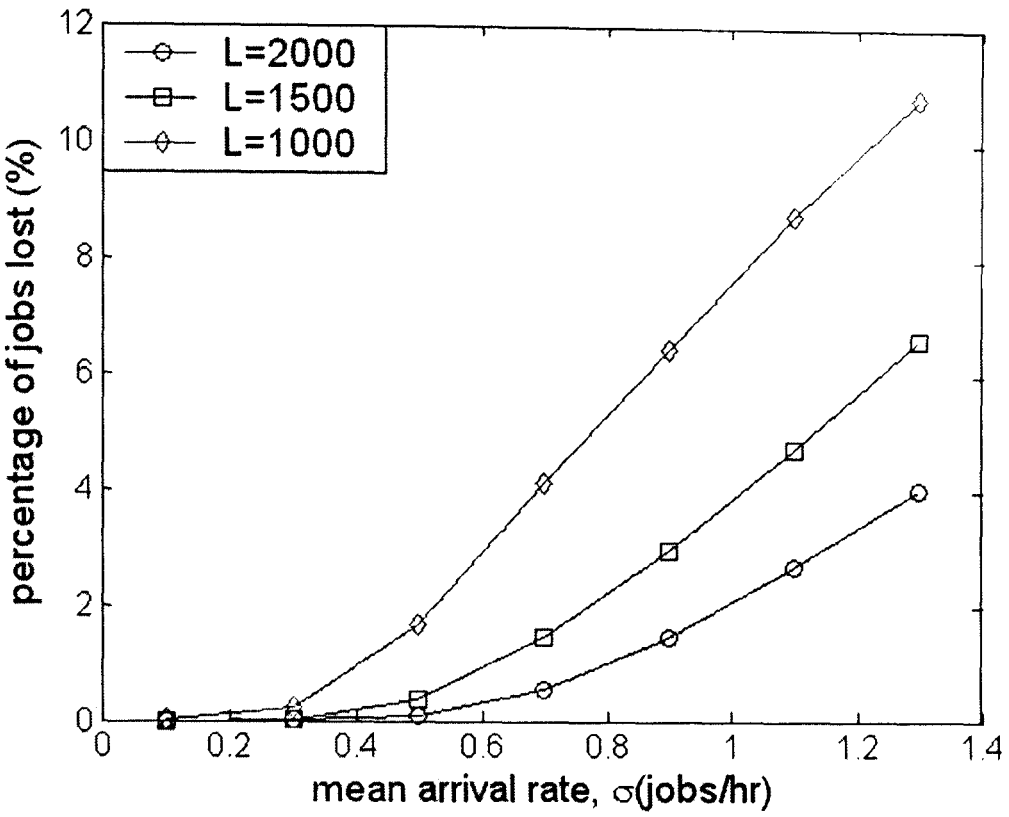

Figure 5.3: PJL as a function of σ for node 1 of IPP model.

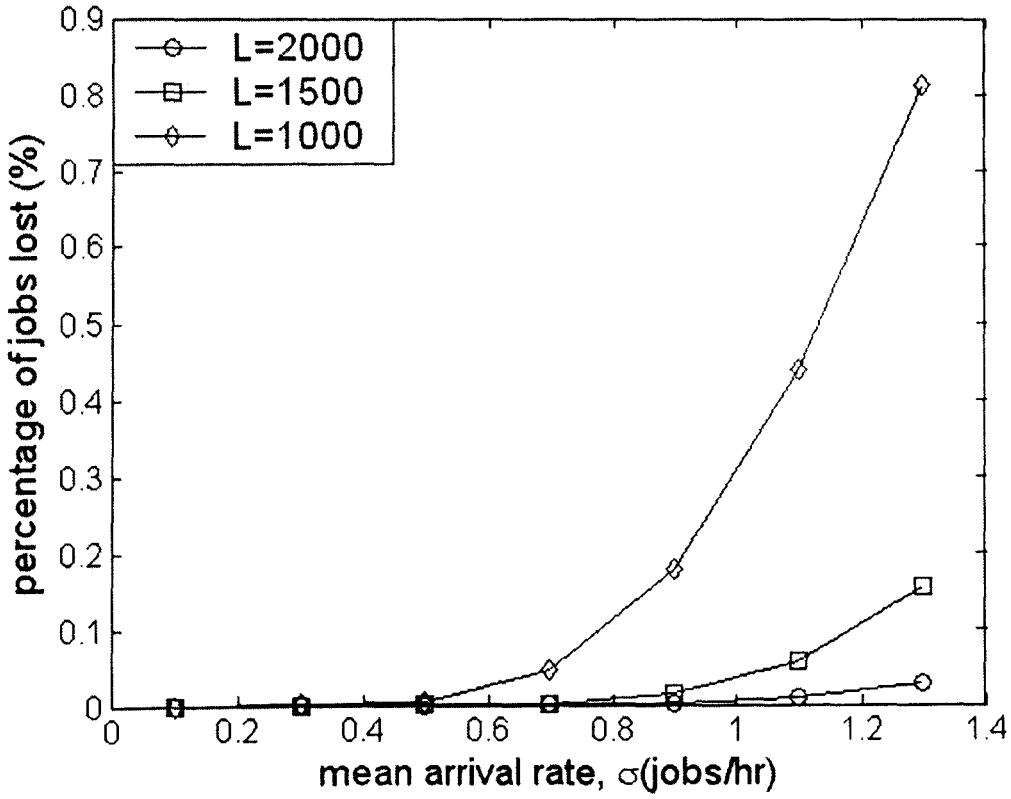

Figure 5.4: PJL as a function of σ for node 2 of IPP model.

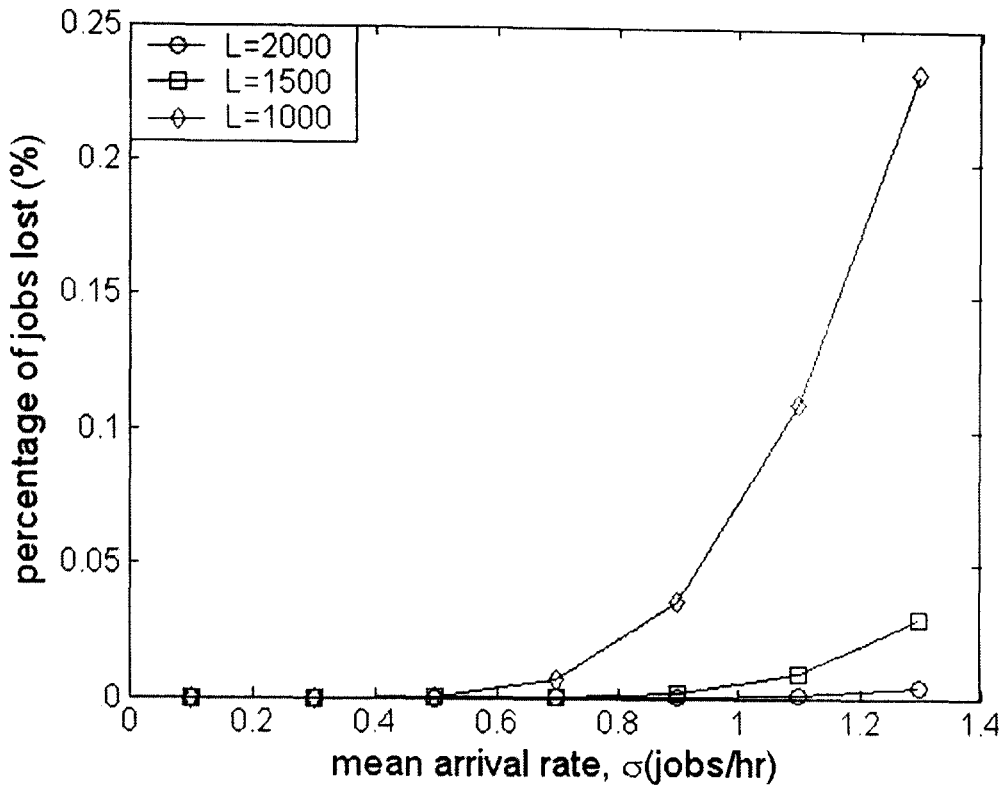

Figure 5.5: PJL as a function of σ for node 3 of IPP model.

In this section a solution technique is presented for performability analysis of open queuing networks with server breakdowns, repairs, external job arrivals and finite queuing capacity where Poisson and IPP have been used for arrival and departure process modelling respectively. Each node has been considered as an $MMPP/M/1/L$ queuing model. An iterative procedure that solves MMPP/M/1/L models in each iteration has been employed and numerical results have been obtained. In addition to the Spectral Expansion method, several other engineering techniques such as probabilistic splitting of the departure process and reconstruction of the arrival processes have been used. The accuracy of the proposed approach is tested together with the joint state approach in the following sections.

5.3 Modelling Open Networks with Breakdowns, Repairs and Finite Buffers, Using Joint State Approach

Since in networks of queues the departed jobs may be fed to other relevant queues, as arrivals for further servicing, it is important to analyse burstiness that can be caused because of the feedbacks from other nodes. This makes departure process modelling a challenging task for open networks. The results presented in the previous section shows that even in case of

finite queuing capacities, having bursty arrivals caused by feedbacks, can affect the system's performance significantly.

The joint state approach is essentially the Markov-modulated systems approach, in which the whole system is conceived as a Markov-modulated one. Joint state approach uses a binary random variable $O_k(t)$ which is equal to zero in case the server is broken and one in case the server is operative. Then the joint state of all the servers for an open network with K nodes is an ordered K-bit binary random variable, $(O_1, O_2, \ldots, O_K)$. The total number of modulating phases or modulating states is 2^K.

In this approach it is assumed that the total job arrivals and total job departure processes at any node during any given modulating phase, is Poisson. Such an assumption used in (Chakka and Mitrani 1996) together with a fast converging iterative algorithm which was effective to compute steady state solution for open queuing networks with unbounded queuing capacities. Similar approaches are given in this chapter for the case where each node in the network has queues with finite capacities.

In the joint state approach each node k ($k = 1, 2, \ldots, K$) is taken as Markov-modulated queue or an $MMPP/M/1/L$ correlated queue where the modulating process has 2^K phases, and these phases are synonymous with the joint states of the servers. In phase i the arrival rate to node k is denoted by $\hat{\sigma}_{k,i}$ and the service rate is denoted by $\mu_{k,i}$.

For this system as well, the transition matrix A is used for purely lateral transitions which occur when an operative server breaks down or a broken server becomes operative. In this case each of these transitions occur with rates ξ_k or η_k respectively. For a three node network (given in figure 5.1) the transition matrices are given below. The vectors given before the matrix shows the operative states $O_k(t)$. The same $O_k(t)$ are used for columns as well with the same order.

$$A_j = A = \begin{array}{c} \\ (0,0,0) \\ (0,0,1) \\ (0,1,0) \\ (0,1,1) \\ (1,0,0) \\ (1,0,1) \\ (1,1,0) \\ (1,1,1) \end{array} \begin{array}{c} \begin{array}{cccccccc} (0,0,0) & (0,0,1) & (0,1,0) & (0,1,1) & (1,0,0) & (1,0,1) & (1,1,0) & (1,1,1) \end{array} \\ \begin{pmatrix} 0 & \eta_3 & \eta_2 & 0 & \eta_1 & 0 & 0 & 0 \\ \xi_3 & 0 & 0 & \eta_2 & 0 & \eta_1 & 0 & 0 \\ \xi_2 & 0 & 0 & \eta_3 & 0 & 0 & \eta_1 & 0 \\ 0 & \xi_2 & \xi_3 & 0 & 0 & 0 & 0 & \eta_1 \\ \xi_1 & 0 & 0 & 0 & 0 & \eta_3 & \eta_2 & 0 \\ 0 & \xi_1 & 0 & 0 & \xi_3 & 0 & 0 & \eta_2 \\ 0 & 0 & \xi_1 & 0 & \xi_2 & 0 & 0 & \eta_3 \\ 0 & 0 & 0 & \xi_1 & 0 & \xi_2 & \xi_3 & 0 \end{pmatrix} \end{array}$$

$B_j = B$ for $j = 0, 1, \ldots, L$;

$B = Diag[\hat{\sigma}_{k,0}, \hat{\sigma}_{k,1}, \ldots, \hat{\sigma}_{k,N}]$
$C = Diag[\mu_{k,0}, \mu_{k,1}, \ldots, \mu_{k,N}]$

$C_j = C$ for $j \geq 0$, $C_0 = 0$.

Once the steady state probabilities, $P_{i,j}$ are obtained, the total rate of job departure process $\hat{\nu}_{k,i}$ and $\hat{\sigma}_{k,j}$ can be computed from equations 2.2 and 2.3 respectively. Iterative solution given in section 2.3.4 is employed in order to compute the mean queue length values at each node in the system by using these equations.

Since the solution given is approximate, it is important to address the accuracy of the models proposed. For this purpose, various simulation results have been obtained for a three node open network with various queuing capacities. The results are presented comparatively for both IPP departure modelling and joint state approaches in the following section.

5.4 Evaluation of the Accuracy of IPP Departure and Joint State Models

In this section numerical results are presented for three server open networks with breakdowns and repairs. Since the solution given is approximate, it is important to address the accuracy of the models proposed. Various simulation results have been obtained for a 3-node open network with various queuing capacities for this purpose. The simulation results are presented comparatively with results of IPP and joint state approaches. The tolerance value ϵ is taken as 0.01 in all computations. Service rates, failure rates and repair rates of nodes 1, 2, and 3 are $\mu_1 = 5.0$ jobs/hr, $\xi_1 = 0.001$/hr, $\eta_1 = 0.02$/hr, $\mu_2 = 6.0$ jobs/hr, $\xi_2 = 0.002$/hr, $\eta_2 = 0.03$/hr, $\mu_3 = 7.0$ jobs/hr, $\xi_3 = 0.003$/hr, $\eta_3 = 0.05$/hr respectively. Since the most challenging systems are the ones which have external arrivals to only one of the servers as shown in (Chakka 1995, Chakka and Mitrani 1996) the arrival rates are taken as $\sigma_2 = \sigma_3 = 0$.

The routing probability matrices Q^1, and Q^2 are given as:

$$Q^1 = \begin{pmatrix} 0 & 0.4 & 0.6 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 1.0 \end{pmatrix}, \quad Q^2 = \begin{pmatrix} 0 & 0.4 & 0.6 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0.5 & 0 & 0 & 0.5 \end{pmatrix}$$

Figures 5.6-5.17 show simulation results together with results obtained using the analytical models proposed. The buffer sizes are equal and noted as $L_k = L$, $k = 1, 2, \ldots, K$ in the figures 5.6-5.17. Routing probability matrix Q^1 is used for figures 5.6-5.11.

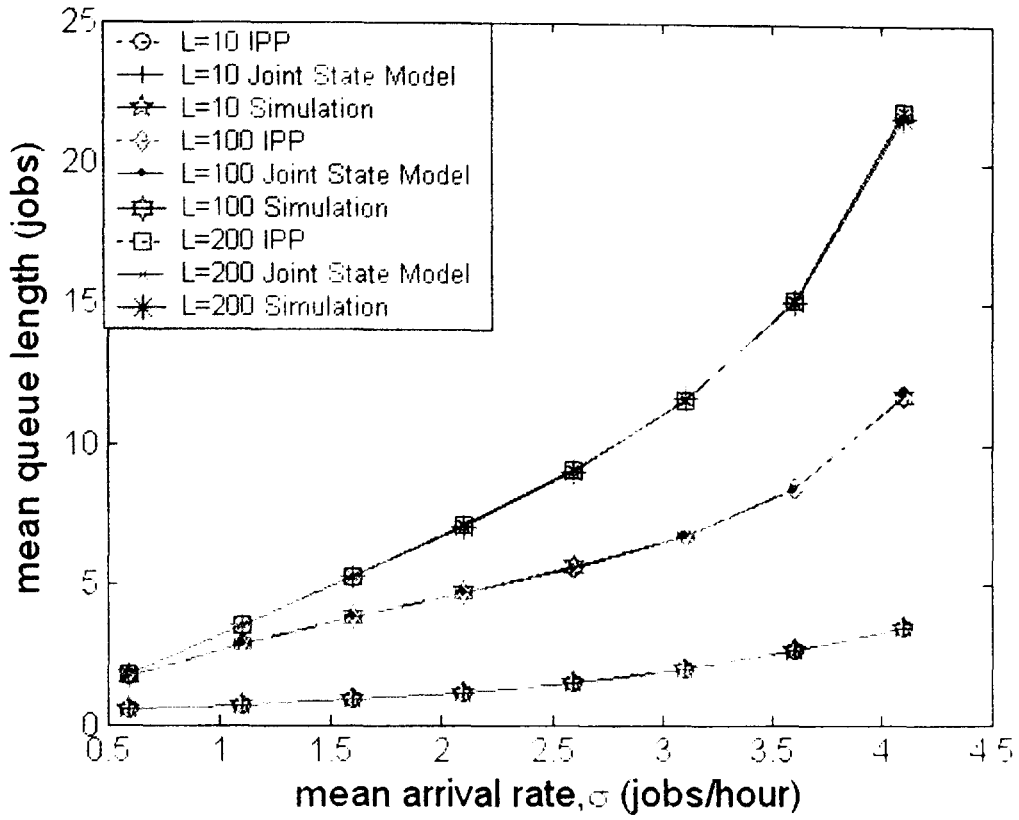

Figure 5.6: MQL for node 1, Q^1 and small L values.

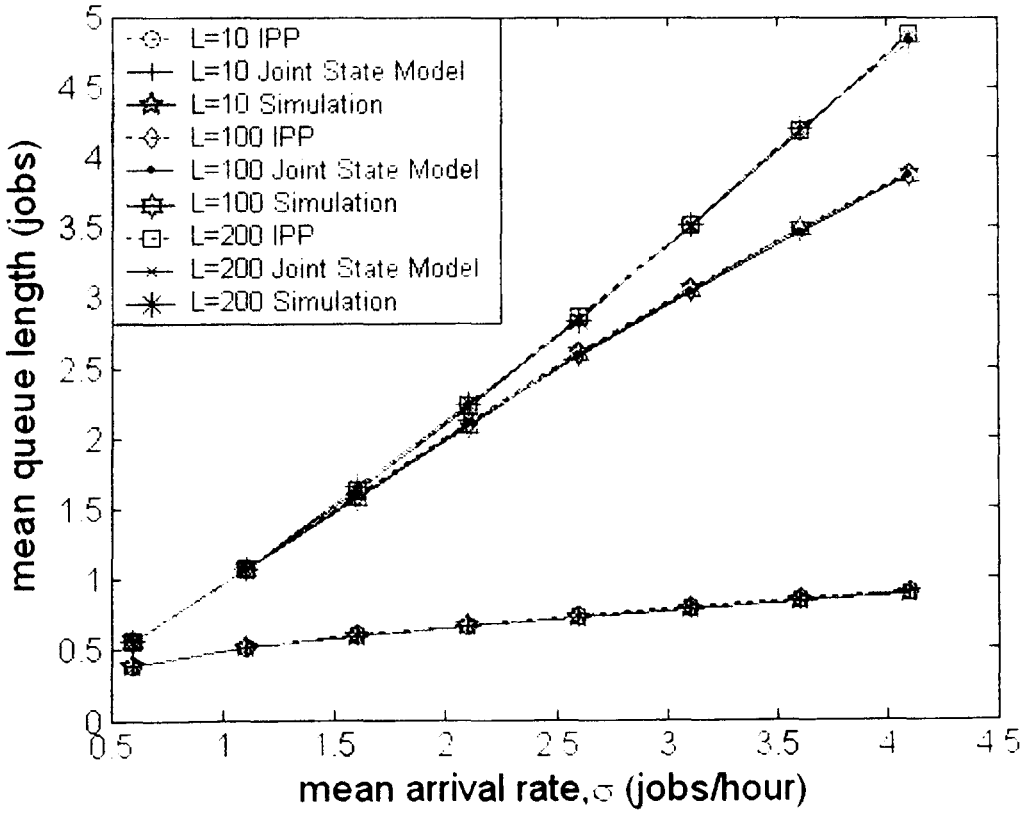

Figure 5.7: MQL for node 2, Q^1 and small L values.

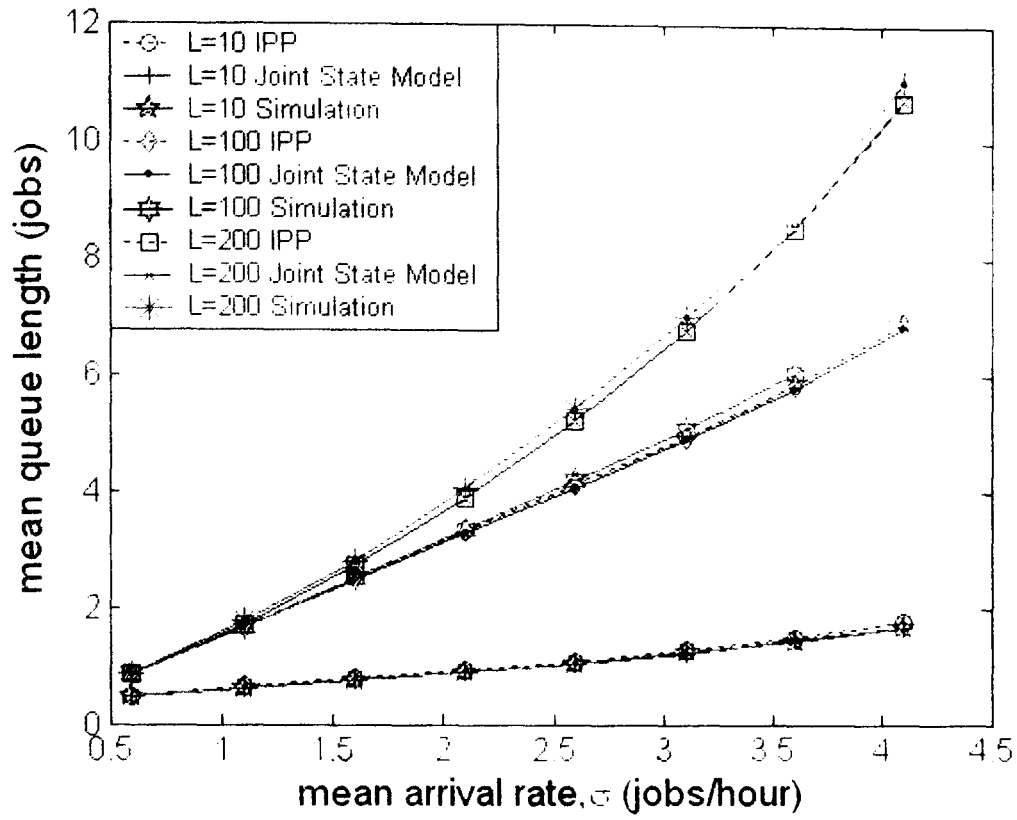

Figure 5.8: MQL for node 3, Q^1 and small L values.

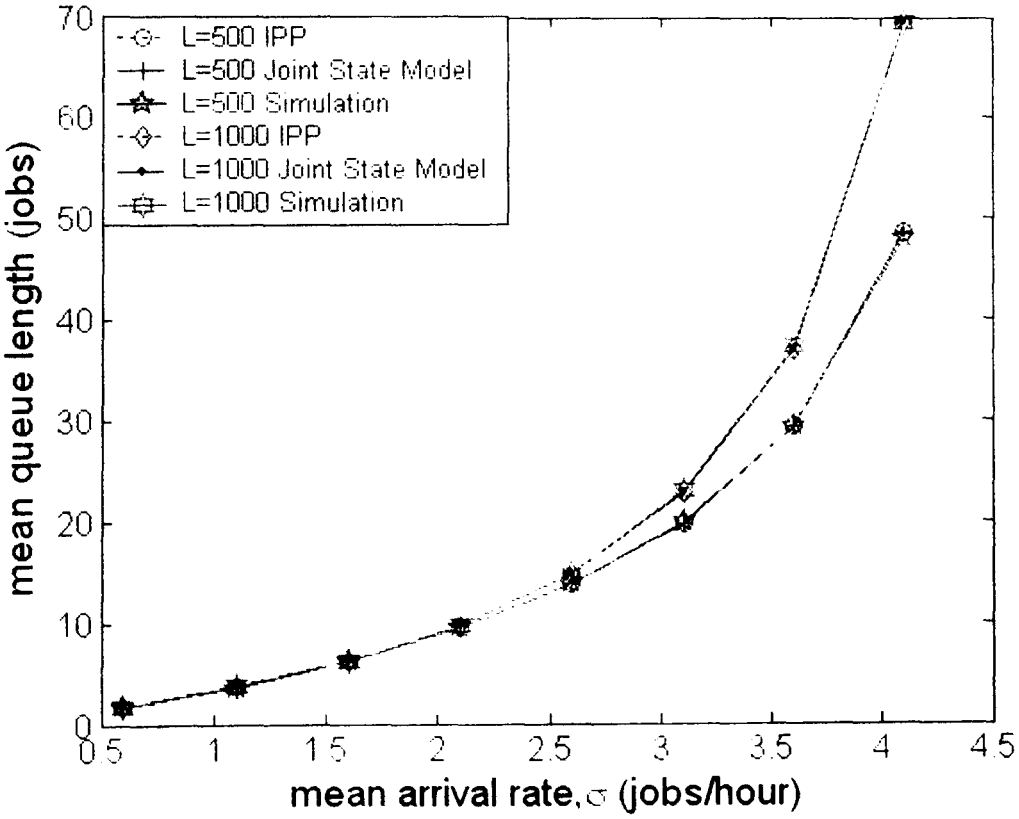

Figure 5.9: MQL for node 1, Q^1 and large L values.

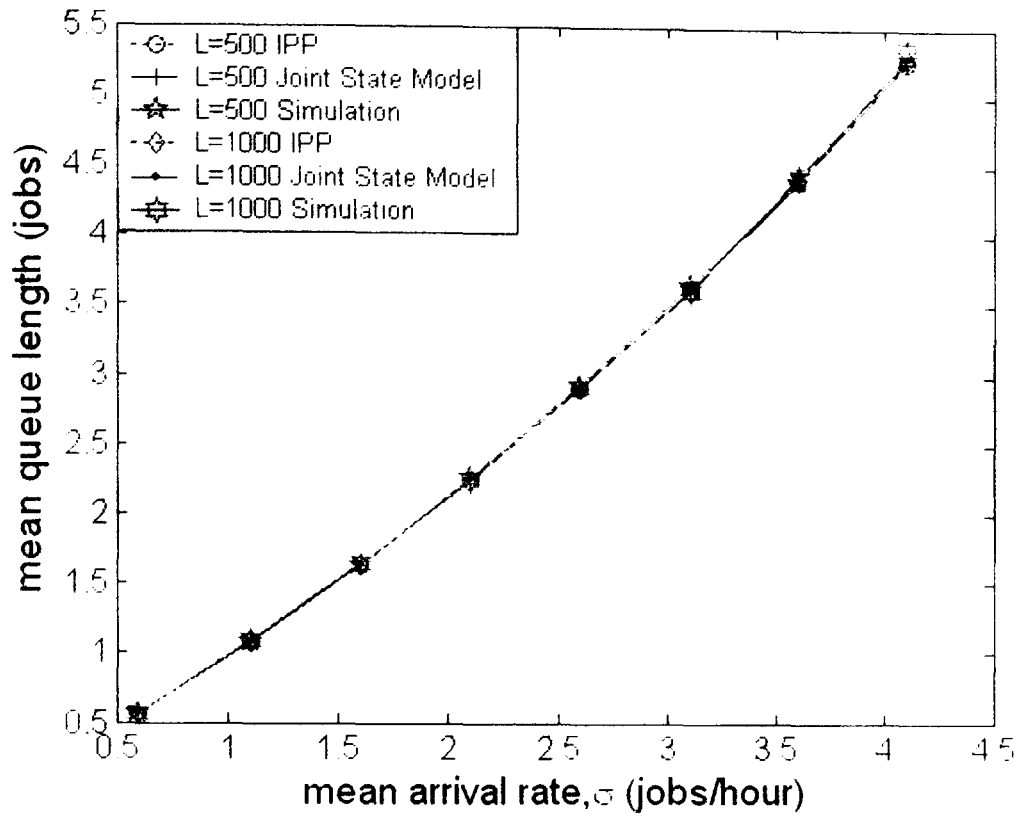

Figure 5.10: MQL for node 2, Q^1 and large L values.

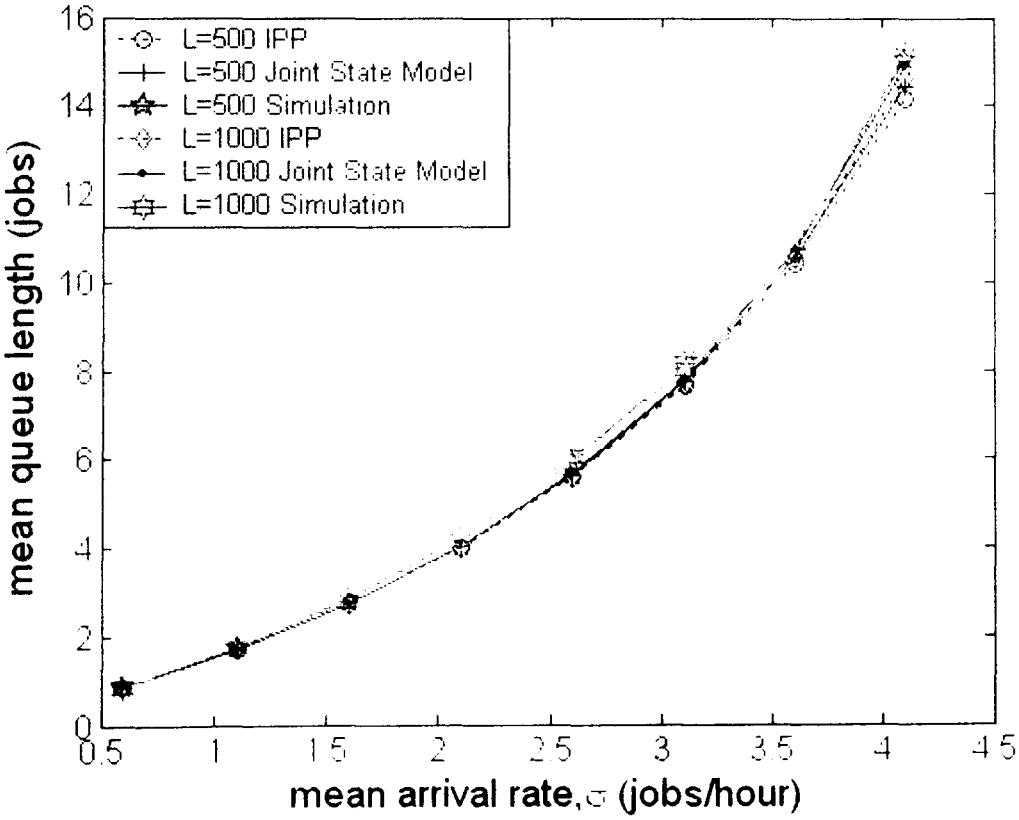

Figure 5.11: MQL for node 3, Q^1 and large L values.

The figures 5.6-5.11 show that for feed forward networks (i.e. when the routing probability matrix Q^1 is used), both IPP and joint state models give close approximations. It can clearly be seen that the results are closer for node 1 of the tandem networks.

In figure 5.10 the results of systems with $L = 500$ and $L = 1000$ overlap. Since the probability of node 2 to receive arrivals is 40% (arrivals are from node 1 only) the MQL values are not affected by the queue capacity.

When the queuing capacity is relatively large, joint state approach gives marginally better approximations than the IPP model. Especially when figure 5.11 is considered the results show that for node 3, the joint state model shows better approximations then IPP model when high arrival rates are expected.

Routing probability matrix Q^2 is used for figures 5.12-5.17. Results are obtained for open networks with feedbacks.

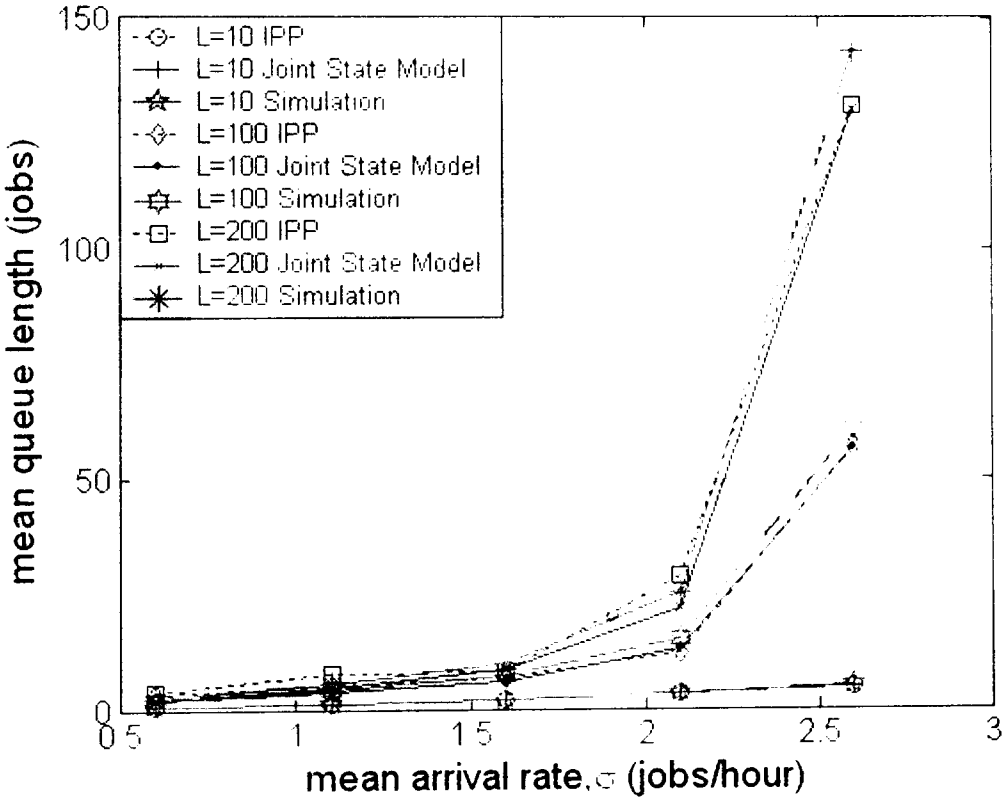

Figure 5.12: MQL for node 1, Q^2 and small L values.

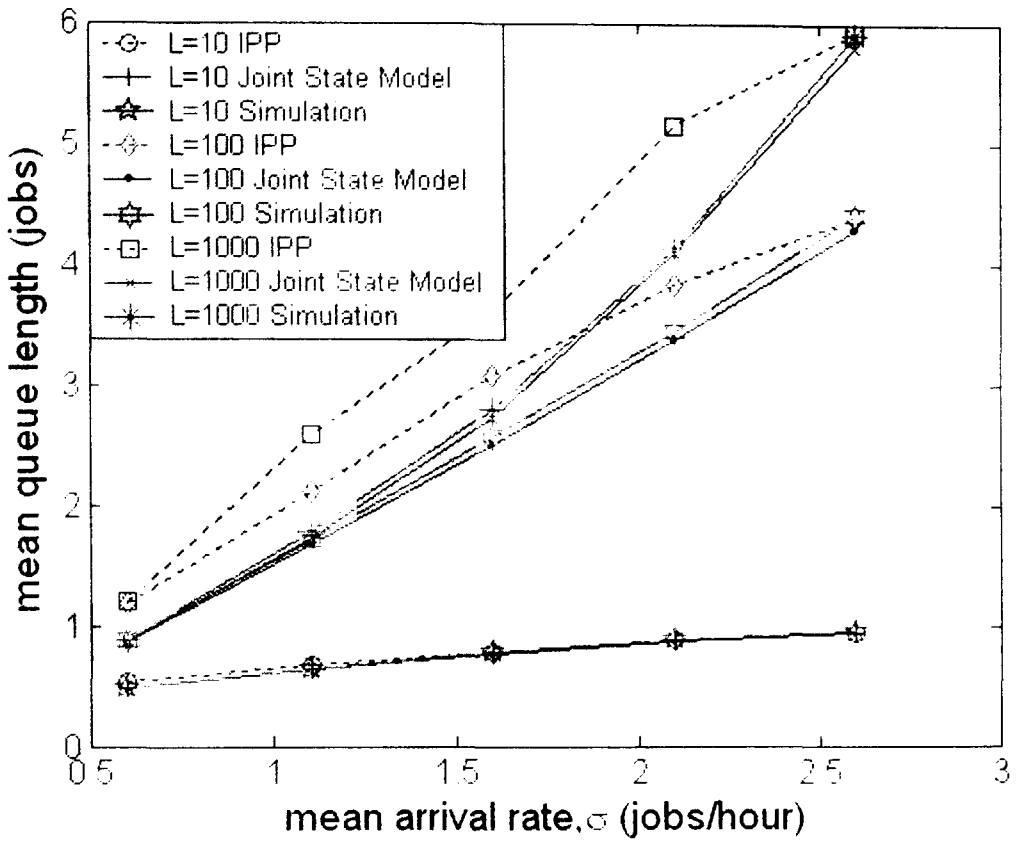

Figure 5.13: MQL for node 2, Q^2 and small L values.

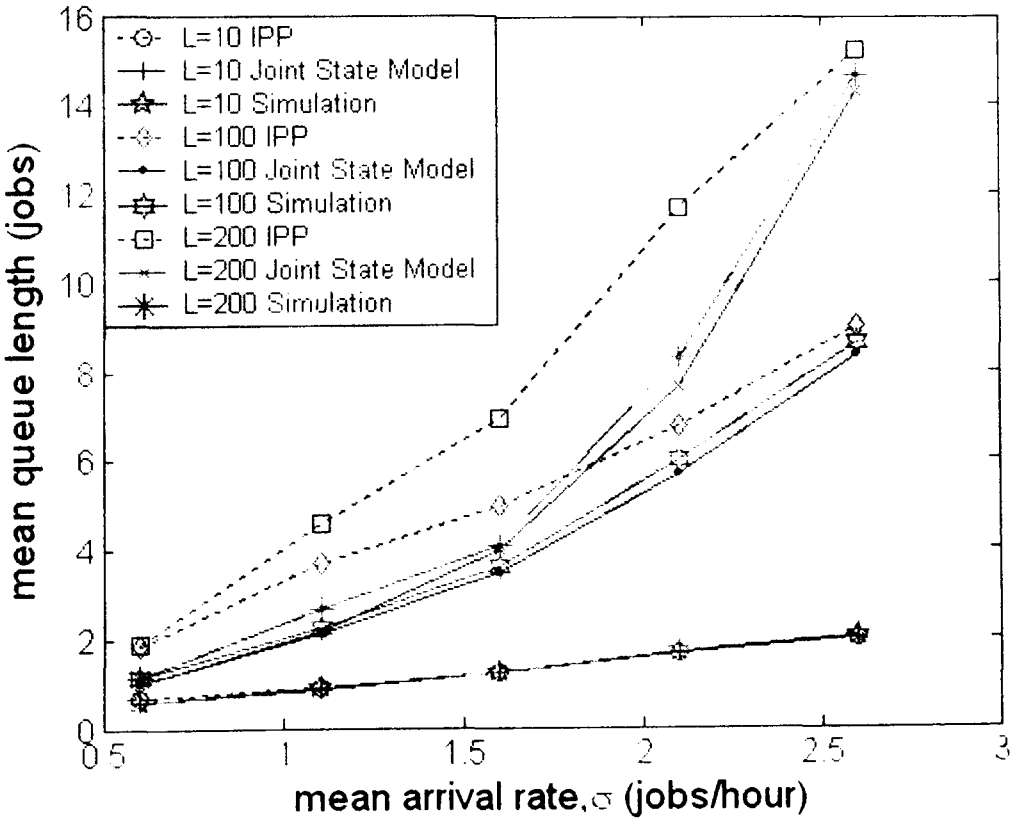

Figure 5.14: MQL for node 3, Q^2 and small L values.

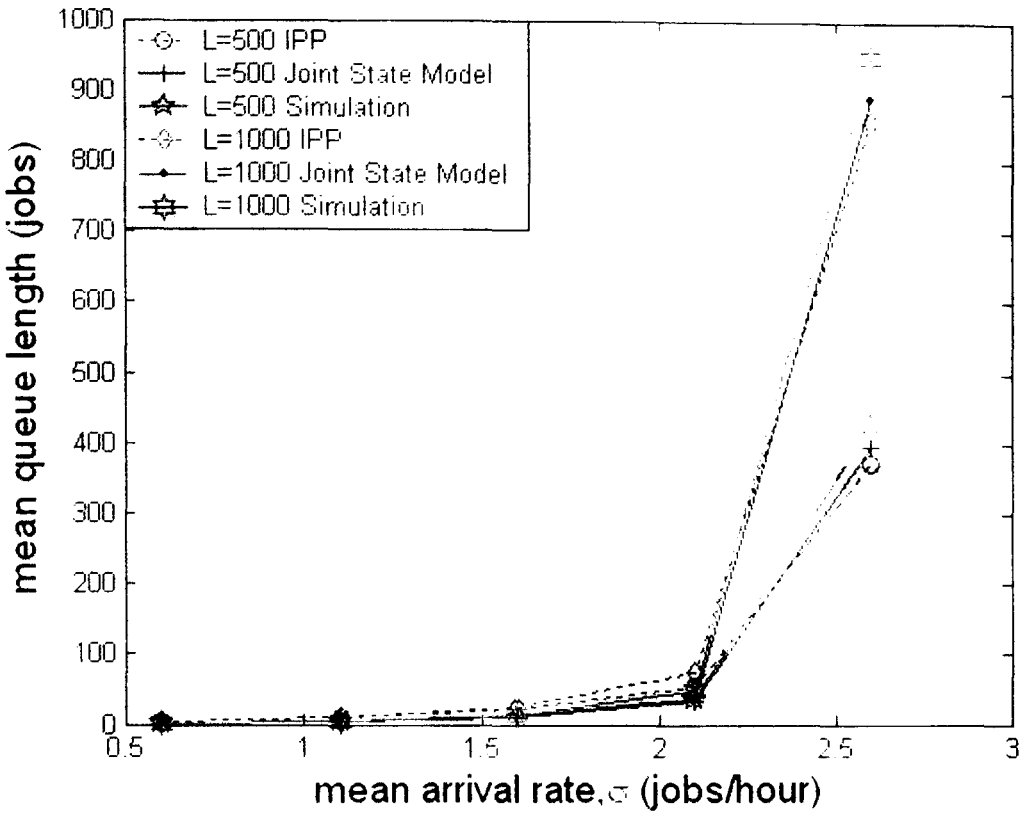

Figure 5.15: MQL for node 1, Q^2 and large L values.

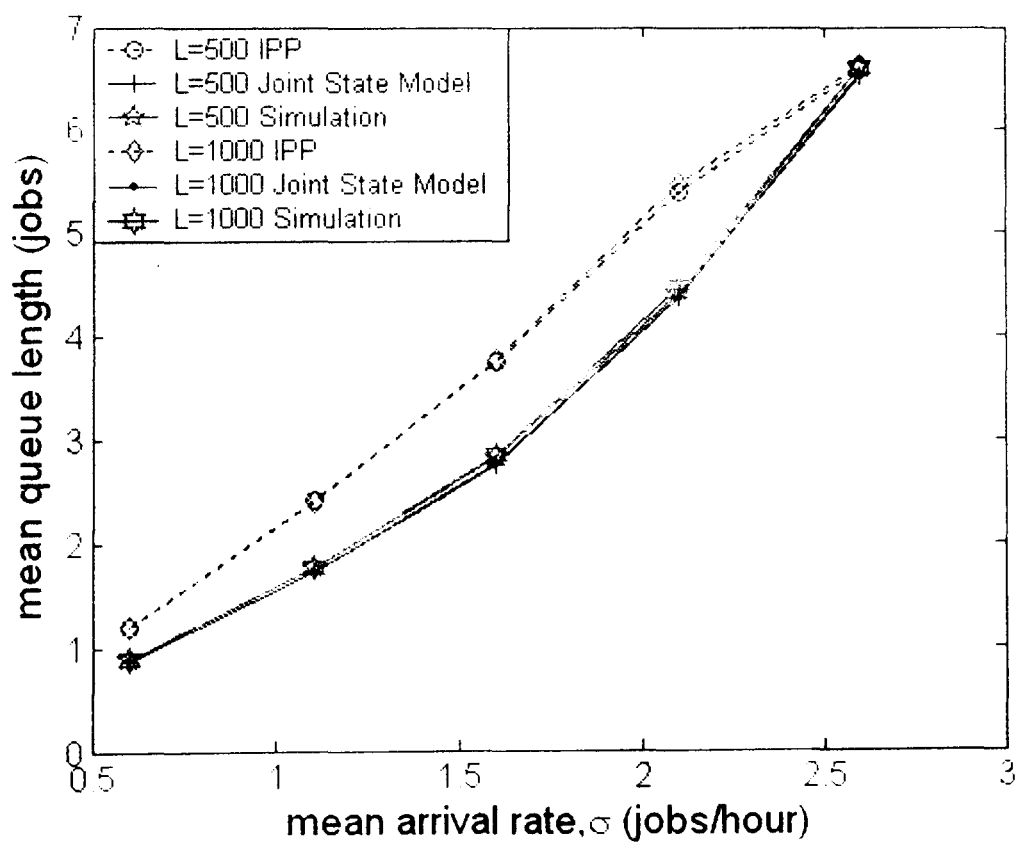

Figure 5.16: MQL for node 2, Q^2 and large L values.

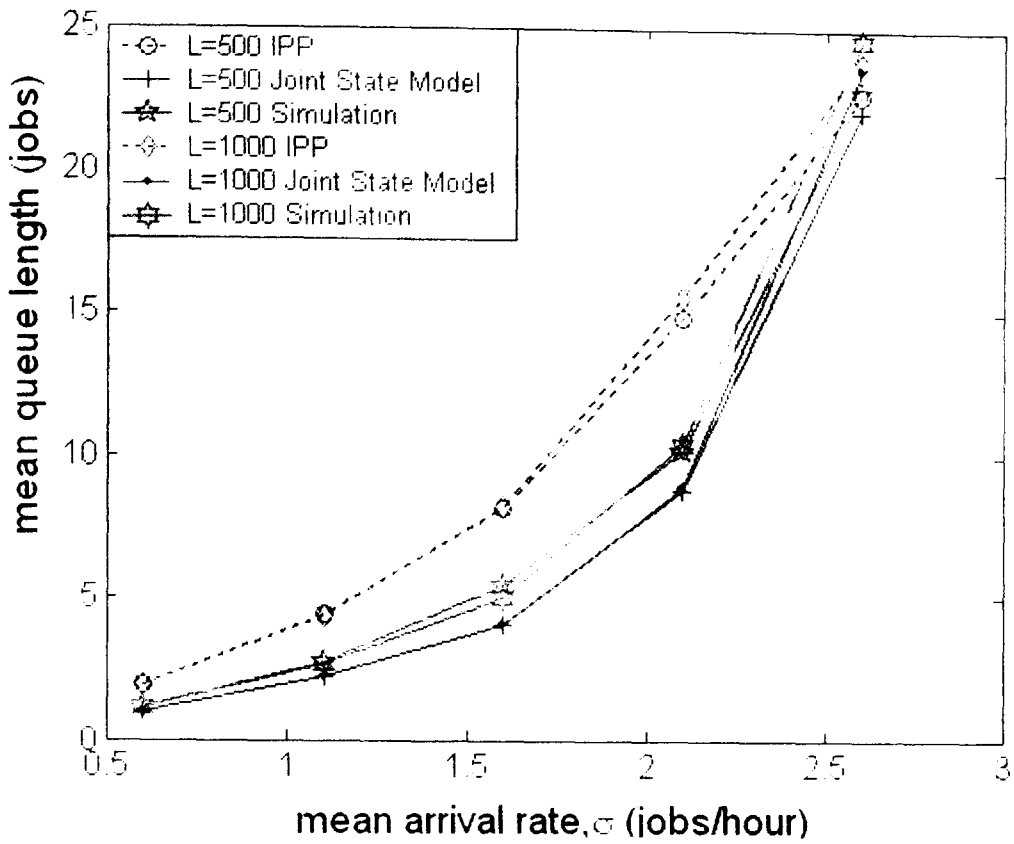

Figure 5.17: MQL for node 3, Q^2 and large L values.

The figures 5.12-5.17 show that for networks with feedbacks (i.e. when the routing probability matrix Q^2 is used), IPP model becomes less accurate. IPP departure model works well for feed forward networks but in case of feedback networks the accuracy of IPP approach decreases especially for nodes two and three. Since IPP is a renewal process but feedback results in non-renewal arrivals, the use of IPP model for open networks with feedbacks is not favourable. In contrast, joint state model gives close approximations for open networks with feedbacks as well.

In the following computations open networks with heterogeneous queue sizes are considered. Numerical results are given for open networks with routing probability matrices Q^1 and Q^2. The accuracy of IPP and joint state approaches are evaluated for a three node system with different queue sizes at each node. The queue capacities for nodes 1, 2, 3 are taken as $L_1 = 200$, $L_2 = 10$, and $L_3 = 50$ respectively. All the other parameters are taken same as the previous computations. The results are illustrated in figures 5.18-5.21

The results are similar to the results of previous computations. In case of feed forward systems both IPP and joint state approaches perform well for systems with heterogeneous queue capacities as well. However, when systems with feedbacks are considered, the IPP approach does not perform well, whereas joint state approach still gives good approximations.

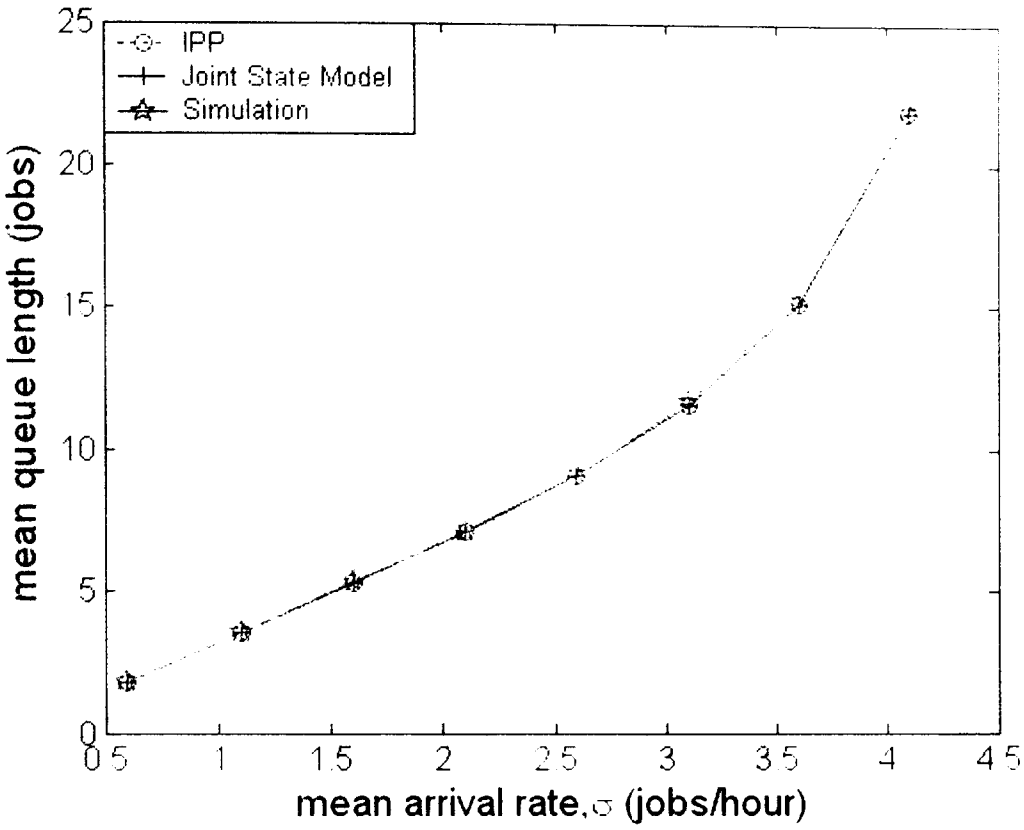

Figure 5.18: MQL for node 1, of open network with heterogeneous queue capacities and Q^1.

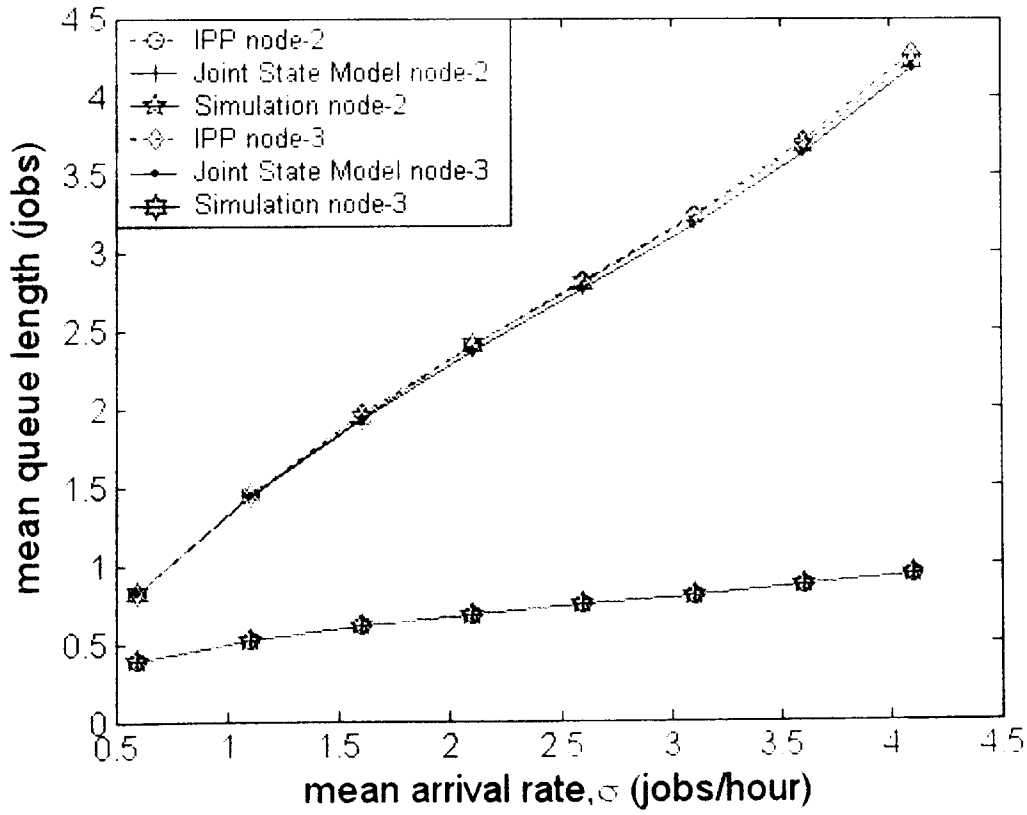

Figure 5.19: MQL for nodes 2 and 3, of open network with heterogeneous queue capacities and Q^1.

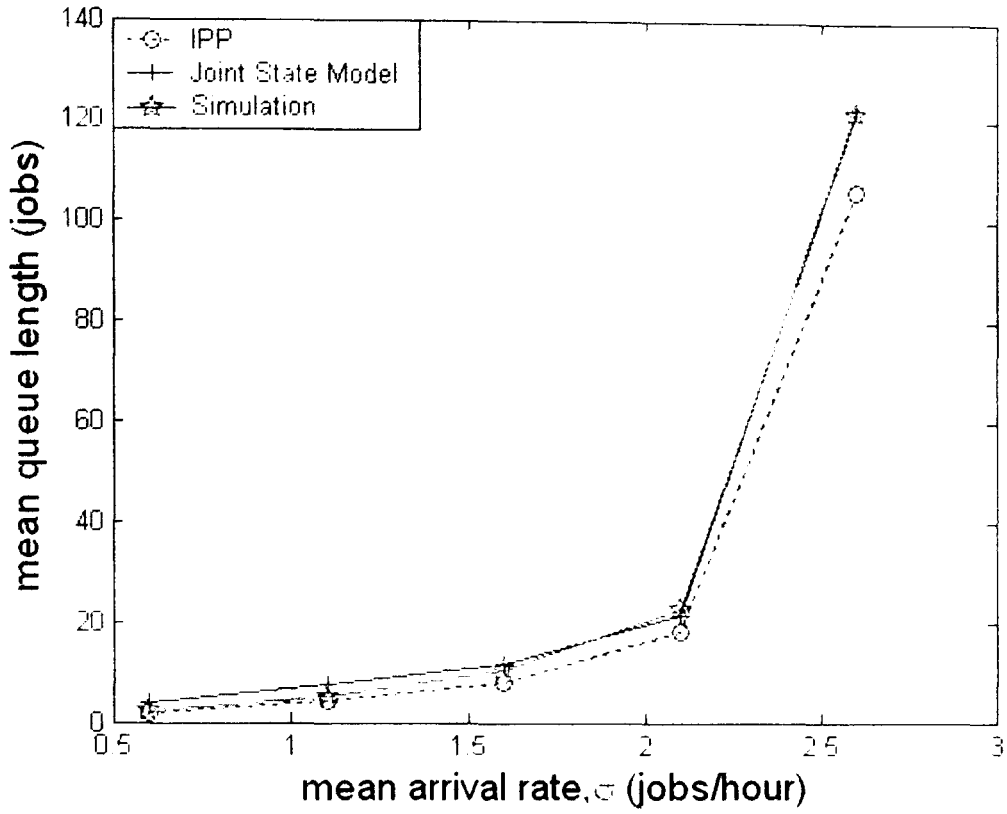

Figure 5.20: MQL for node 1, of open network with heterogeneous queue capacities and Q^2.

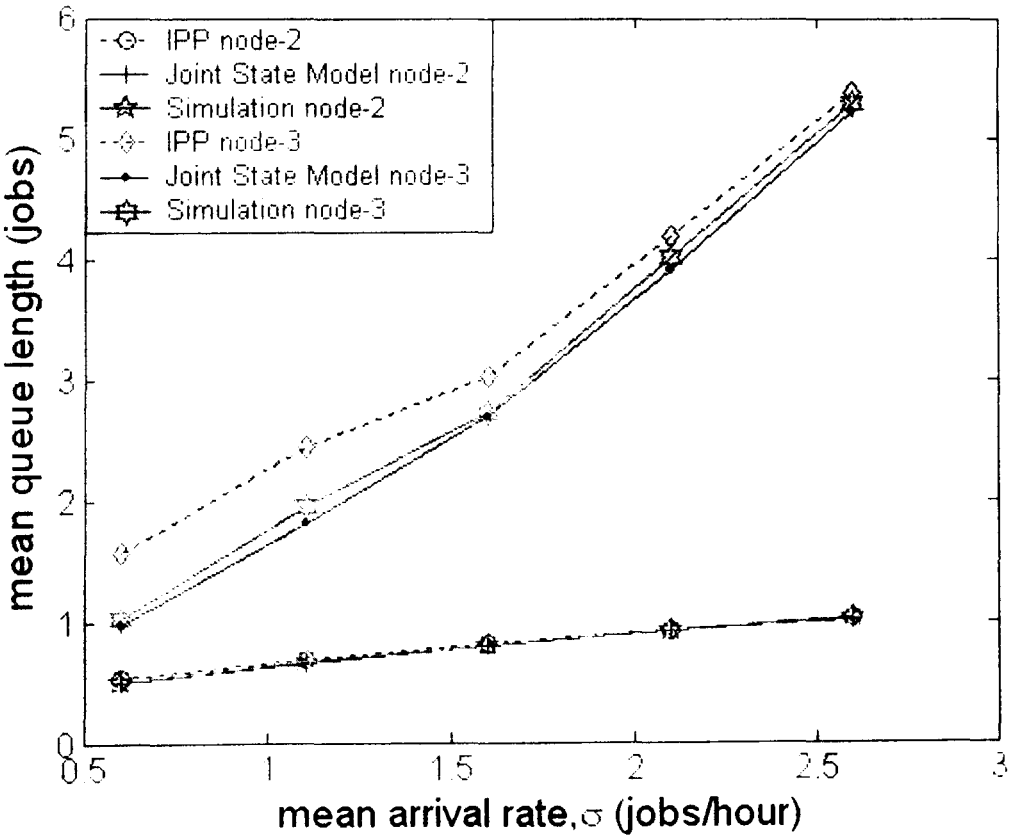

Figure 5.21: MQL for nodes 2 and 3, of open network with heterogeneous queue capacities and Q^2.

5.5 Conclusion

In this chapter models and fast solution techniques are presented for performability analysis of open queuing networks with server breakdowns, repairs, external job arrivals and finite queuing capacity. In the first approach MMPP and IPP have been used for overall arrival at a node and departure process from a node, respectively. Each node has been considered as a MMPP/M/1/L queuing model. MMPP/M/1/L models have been analysed and numerical results have been presented. Other than the Spectral Expansion method, several other engineering techniques have been used in order to compute the probabilistic splitting of the departure process and reconstruction of the arrival processes. In the second approach, the joint state model has been used for both arrival and departure process modelling. The transition matrices are derived by using the random variables that are defined to represent the joint states used. Finally, numerical results have been presented for various systems. Simulation results are also presented for validation and comparison purposes.

The proposed IPP and joint state models are applicable for open networks with bounded queues. The approaches are reasonably accurate for feed forward open networks. However, in the case of the open network systems with feedbacks the IPP model does not perform well. All three stages give good approximations for both cases, when the joint state approach is employed.

Numerical results are obtained for a three node system with heterogeneous queuing capacities as well. The IPP model and joint state model are compared. Similar to open network systems where the queue capacity is same for all nodes, the IPP model does not perform well for networks with feedbacks. However, the joint state approach provides far better approximations.

Since IPP is a renewal process but feedbacks cause non-renewal overall arrivals, joint state approach performs better than IPP approach for open network systems with feed backs and bounded queues.

Chapter 6

Performability Analysis of Homogeneous, and Highly Available Farm Paradigm, Multi-server Systems with Breakdowns and Deferred Repairs

Fault-tolerant systems with repair-upon-failure strategy can become expensive in terms of labour, time and cost. Deferred repair strategies are effectively being used in order to reduce these expenses (Carrasco 2006). While postponing these repairs, it is essential to keep the whole system capable of dealing with user requests. For this purpose, usually, a threshold value is defined. When the number of operative servers becomes the predefined threshold value because of the failures, a repairman is called (Tang and Trivedi 2004, Sun et. al. 2005). New generation high-availability commercial computer systems incorporate deferred repair service strategies and pure availability metrics may no longer present a sufficient amount of information to evaluate these systems (Tang and Trivedi 2004, Sun et. al. 2005). Homogeneous multi-server systems, and highly available multiprocessor systems with one head and several computation nodes are common configurations. Deferred repairs can be used for such systems for reducing repair costs when no permanent repair facility exists on premises. Performability evaluation of such systems is very important since the systems are fault-tolerant and deferred repairs affect the system significantly in terms of availability. In this chapter, performability modelling of homogeneous, and highly available farm paradigm, multi-server systems is presented. To account for various types of delays, such systems are

modelled and solved for exact performability measures for both bounded and unbounded queuing systems where a deferred repair strategy is considered.

6.1 Using Deferred Repairs to Reduce the Cost

Deferred repairs are a practice of allowing systems to serve in a degraded mode. The repairs which are prudent but not essential for the systems are postponed. In previous chapters, multi-server systems with traditional repair strategies are considered. A repair is requested as soon as a component fails in order to maximize the performability of the system. However, as a system's complexity and the number of components in the system increases, traditional repair strategies may become expensive especially in terms of time and labour (Sun et. al. 2005, Carrasco 2006). An alternative repair strategy is to defer repairs, and leave the failed components in the system without repair until the number of failed components exceeds a predefined threshold (Bossen et. al. 2002, Tang and Trivedi 2004, Sun et. al. 2005, Carrasco 2006).

The systems using the traditional repair-upon-failure technique have higher availability than the systems with deferred repair strategies, and this affects the performability of the system as well. However, it is possible to reduce the costs by postponing repairs particularly for the systems for which the replacement of failed components is an expensive procedure (e.g. the system is at a remote location). Also, since the costs of components such as memory chips and disks are decreasing, it is affordable to introduce greater number of components to the systems. This makes it possible to tolerate multiple component failures without repair, before the number of failed components reaches a threshold or a scheduled maintenance action occurs (Tang and Trivedi 2004). Both downtime and cost can be reduced with this design approach.

If the computing facilities of the system considered are homogeneous and all servers employed have identical computing responsibilities, the performance of the overall system can become insufficient because of degraded service power caused by failures. For such systems it is possible to defer server repairs while the whole system is still able to handle the incoming job requests (system is still able to provide an acceptable level of service power).

Performance and availability analysis of systems with deferred repairs is useful for the development and optimisation of highly available computer server, storage, and networking systems. In (Temsamani and Carrasco 2002) split regenerative randomisation method is introduced for the transient analysis of a class of Continuous Time Markov Chains, where CTMCs are used to present reliability models of repairable fault-tolerant systems with de-

ferred repairs. Carrasco used a new importance sampling scheme for the efficient simulation of CTMC models used for fault-tolerant systems with deferred repairs in (Carrasco 2006). In (Tang and Trivedi 2004) a hierarchical modelling approach is followed in order to obtain performability measures for systems incorporating deferred repairs. Another approach is presented to optimise the repair strategy for multi-processor systems with deferred repairs in (Sun et. al. 2005) using tools such as RAScad and SHARPE. For performability analysis of multi-server queuing systems, it is important to consider queuing characteristics such as finite and infinite queuing capacities, the mean queue length of the systems considered, and the mean number of jobs lost because of the limited queuing capacities.

In this chapter, analytical modelling for performability evaluation of homogeneous and highly available clusters with breakdowns and deferred repairs is presented. Queuing systems with finite and infinite queuing capacities are considered. For this purpose, the resulting QBD process in the performance models of multiprocessor systems with breakdowns and repair strategies in previous chapters, and (Trivedi et. al. 1990, Chakka et. al. 2002), are extended.

6.2 Homogeneous Multi-Server Systems with Deferred Repairs

The computing facilities of the homogeneous systems are identical and all servers employed have the same computing responsibilities. In such systems, the performance of the overall system can become insufficient because of the degraded service power caused by the accumulation of failures. Since there is no control hierarchy (e.g. no head node which makes the whole system down in case of breakdowns) postponing non essential repairs is easier. It is possible to reduce the repair costs without affecting the availability of the whole system significantly.

In this section approaches are developed for performability evaluation of homogeneous multi-server systems with deferred repairs. Models are presented for systems with reconfiguration/rebooting delays as well as systems without reconfiguration/rebooting delays. The QBD process given in previous sections are suitably extended in order to handle deferred repairs. Since deferred repairs have implications of degraded performance and availability, it is essential to evaluate the performability of such systems. Also, the performability evaluation of such systems is important for optimisation of various system parameters such as deferral threshold to provide an acceptable performability.

For Markov models of repairable fault-tolerant systems, standard simulation of dependability

measures tends to be expensive due to the rarity of the system failure event, and it is required to use another method for performance and availability evaluation. In (Temsamani and Carrasco 2002) the failure events are taken as rare events. Also in (Carrasco 2006) a *rarity* parameter measuring how small failure transition rates are with respect to repair transition rates is defined. It is assumed that failure transition rates are much smaller than repair transition rates. This corresponds to fault-tolerant systems made up of highly reliable components. Such systems are called balanced fault-tolerant systems.

For homogeneous systems with deferred repairs, simulation results are given comparatively with numerical results obtained for homogeneous systems in order to validate the analytical models presented.

6.2.1 Modelling Homogeneous Multi-Server Systems with Deferred Repair Strategies

The homogeneous multi-server system under study is shown in figure 6.1. The system consists of K identical parallel servers, numbered, 1, 2, ..., K, and a common queue. The queue can be bounded or unbounded.

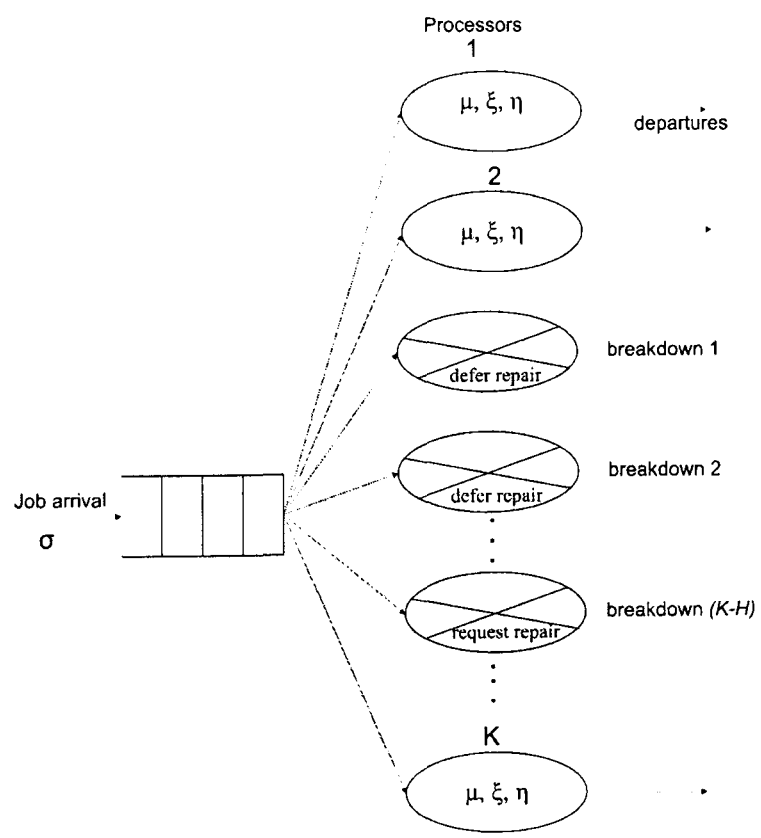

Figure 6.1: A homogeneous multi-server system with deferred repairs.

Jobs arrive at the system in a Poisson stream at a mean rate of σ. Jobs are homogeneous. The service times of jobs served by server k ($k = 1, 2, \ldots, K$) are distributed exponentially with mean $1/\mu$. Server k executes jobs during its operative periods, which are distributed exponentially with mean $1/\xi$. Failed servers are not taken to repair until the number of broken servers reaches a predefined threshold value of $K - H$, where $H < K$. Once the number of broken servers becomes $K - H$ the repairman starts repair work. If the repairman is not present, he/she reaches the location of the system after possible transportation delays in case the system is located in a remote site. The transportation time is given by $1/\eta_t$. If a repairman is present all the servers have the same individual repair time $1/\eta$. The details of two dimensional representation of similar systems are given in previous chapters. Figure 6.2 shows the operative states of this system.

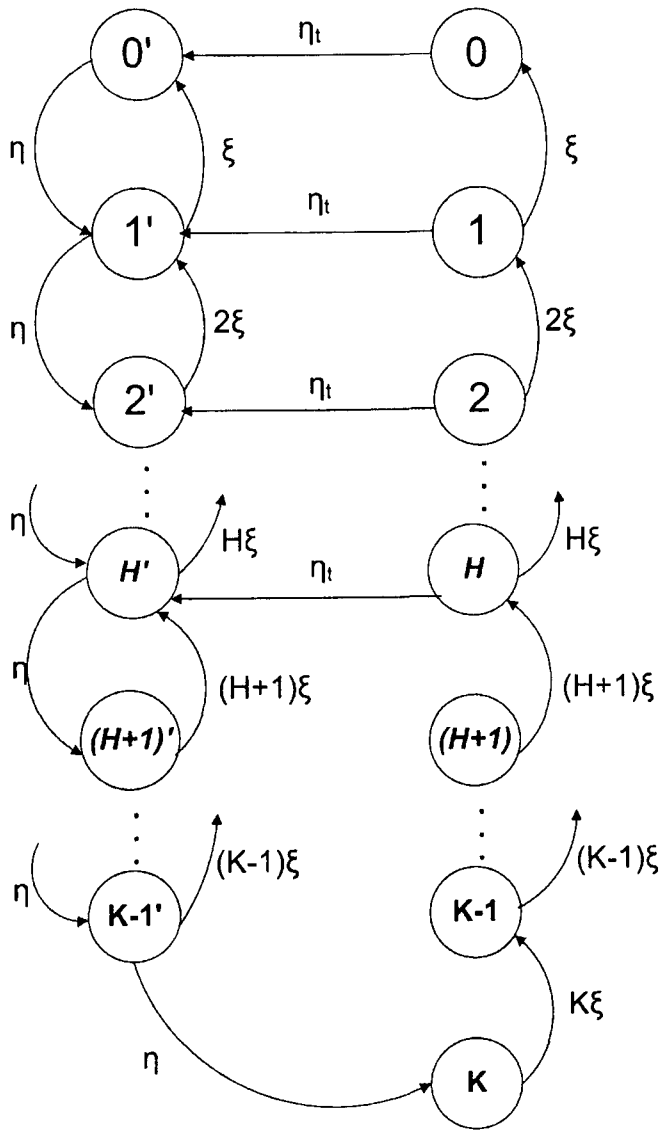

Figure 6.2: Operative states of a homogeneous multi-server system with deferred repairs.

In the Markov chain given, if failures occur while a repairman is present (working on other repairs), he repairs all broken servers until the system is fully operational. No operative server can be idle if there are jobs awaiting service. All inter-arrival, service, operative and repair time random variables are independent of each other. If the number of operative servers is greater than the number of jobs in the system, then the available servers are selected randomly to assign jobs. Services that are interrupted by server breakdowns are eventually resumed.

The states labelled 1, 2, ..., H, $(H+1)$, ..., $K-1$, and K are the K working states of the multi-server system, with that many number of processors in each state. In state 0 all the servers are broken and no repairman is present. The state labelled as H represents the state of system where the number of processors is equal to the threshold value H. The states labelled $0'$, $1'$, $2'$, ..., H', $(H+1)'$, ..., $K-1'$ are the states where, the repair facility is present. It is obvious that the repair facility does not leave the premises until whole system is fully operative. It is a small possibility to have breakdowns before the system is fully repaired but it is not impossible since the servers are put into service once they are repaired. The given model takes these possibilities into account. The total number of states is independent of H and is given as $2K+1$.

The numbering for the transition matrices is done as follows. First $K+1$ numbers (0, 1,...,K) are assigned to states 0, 1, 2, ..., H, $(H+1)$, ..., $K-1$, K and the following K numbers ($K+1$, $K+2$, ..., $2K$) represent states $0'$, $1'$, $2'$, ..., H', $(H+1)'$, ..., $K-1'$. The transition matrices A, B, and C can be given as follows:

$$A_j = A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \eta_t & 0 & 0 & 0 \\ \xi & 0 & 0 & 0 & 0 & 0 & \ddots & 0 & 0 \\ 0 & 2\xi & 0 & 0 & 0 & 0 & 0 & \eta_t & 0 \\ 0 & 0 & \ddots & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & K\xi & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \eta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \xi & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \ddots & 0 & \eta \\ 0 & 0 & 0 & 0 & \eta & 0 & 0 & (K-1)\xi & 0 \end{pmatrix}$$

$B_j = B$ for $j = 0, 1, \ldots, L$;

$B = \text{Diag}[\sigma, \sigma, \ldots, \sigma]$ of size $(2K+1) \times (2K+1)$

$C_j = C$ for $j \geq K$; the threshold $M = K$

$C = \text{Diag}[0, \mu, 2\mu, \ldots, K\mu, 0, \mu, 2\mu, \ldots, (K-1)\mu]$

$C_0 = 0.$

$C_j = \text{Diag}[0, Min\{1,j\}\mu, Min\{2,j\}\mu, \ldots, Min\{K,j\}\mu, 0, Min\{1,j\}\mu, Min\{2,j\}\mu, \ldots, Min\{(K-1),j\}\mu]$ for $1 \leq j < K$.

Using this QBD system, numerical results are presented for homogeneous systems with repair deferrals in the following section.

6.2.2 Numerical Results for Homogeneous Systems with Deferred Repairs

Homogeneous systems with infinite queuing capacities are considered first. In figure 6.3 systems with various number of servers are considered ((a) shows loaded systems, (b) shows systems with relatively lighter loads). Other parameters are given as σ jobs/hour, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hour.

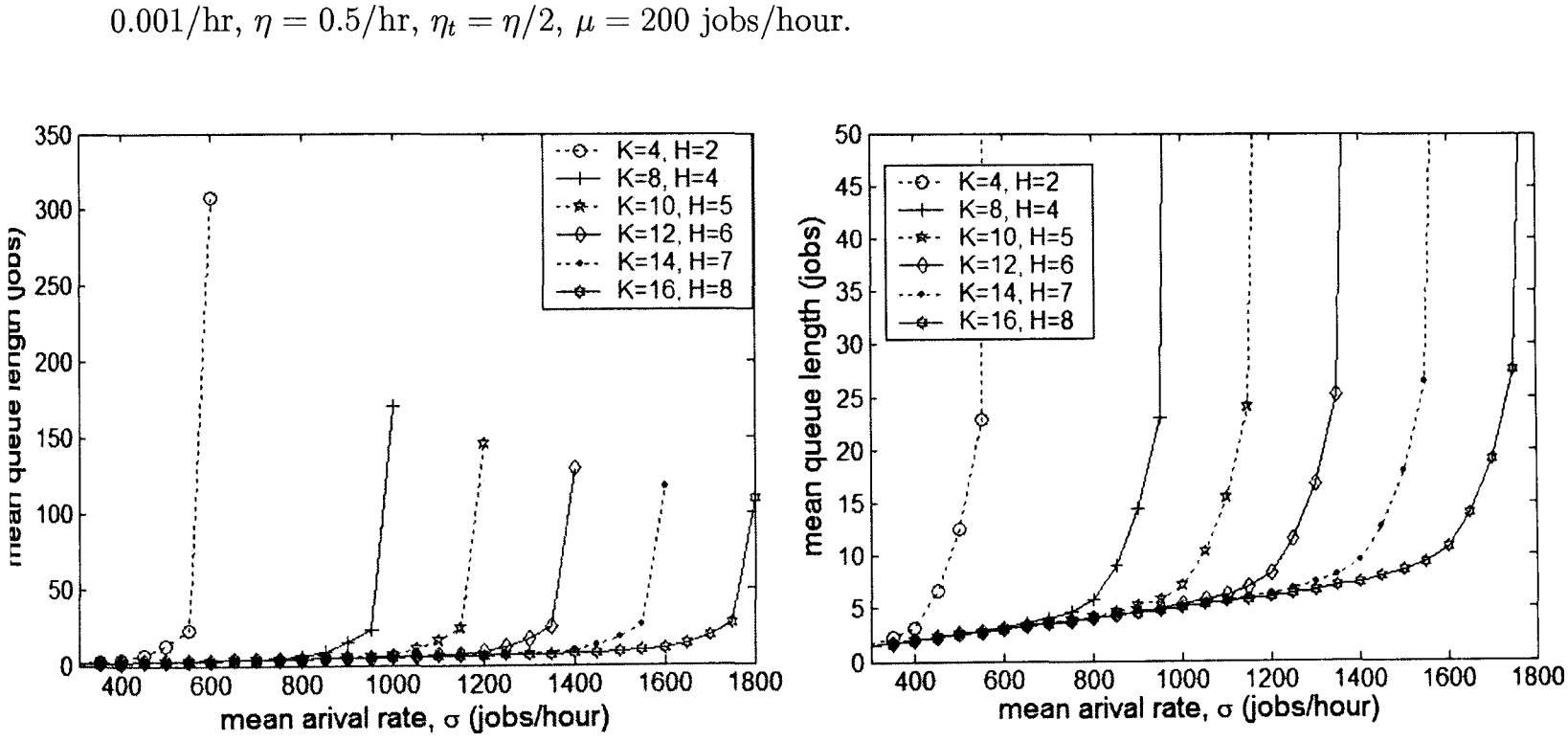

Figure 6.3: MQL as a function of σ and K for homogeneous systems with deferred repairs.

The results given in 6.3 show that systems with larger number of servers have better performability since the threshold value is taken as $K/2$.

A 16 server system is considered in figure 6.4. MQL is computed as a function of arrival rate σ jobs/hour((a) shows loaded systems, (b) shows systems with relatively lighter loads).

Other parameters are, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hour. The results show that systems with greater threshold values perform better as expected. However in case of relatively light arrival rates it can be affordable to defer repairs. It is possible to use the given model for optimisation of the threshold value with respect to cost on performability. Depending on the mean arrival rate expected, users can compute the threshold value which can be used to defer repairs while the systems performability measures are still reasonable.

In figure 6.5 the effects of various transportation delays are shown for systems with various ξ/hr values where $\sigma = 400$ jobs/hour, $\eta = 0.5$/hr, $\mu = 200$ jobs/hr, $H = 2$, and K=4. The results show how the effects of η_t increase as the failure rate of the system increases.

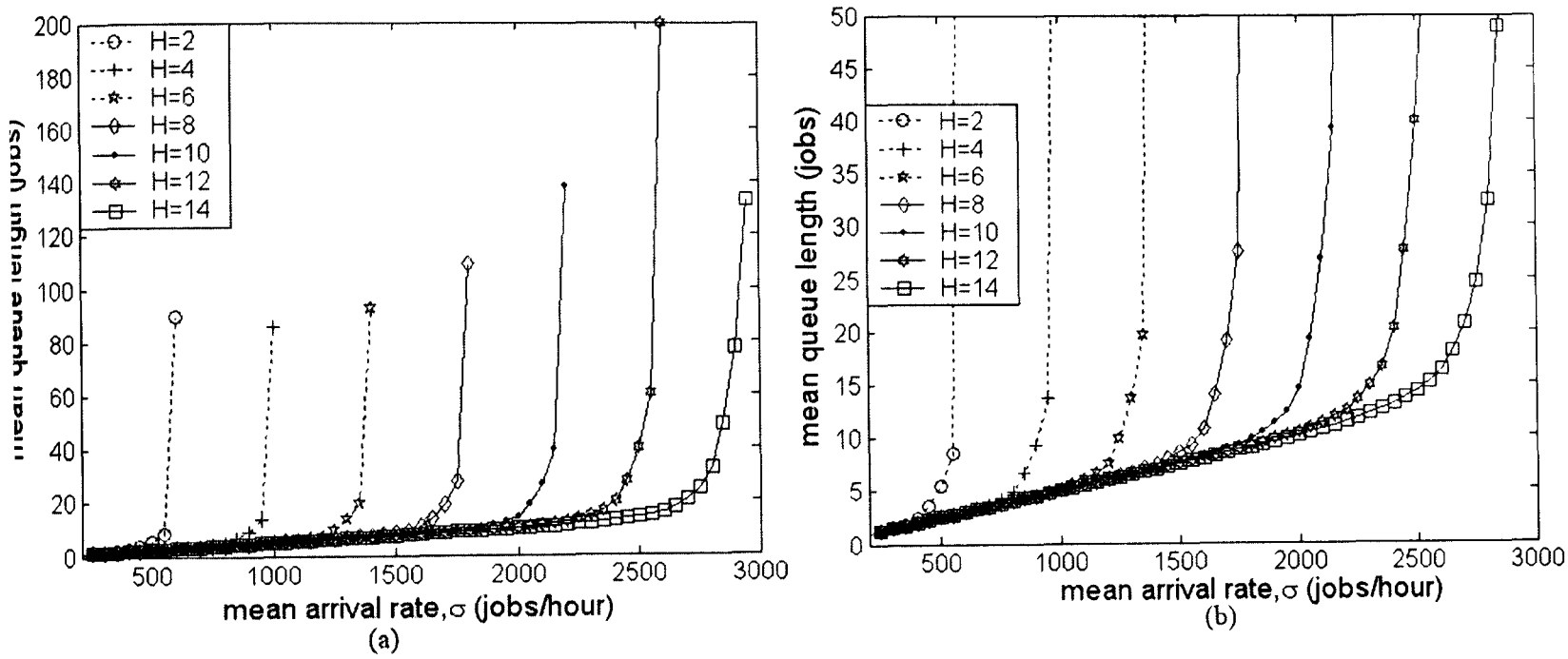

Figure 6.4: MQL as a function of σ for homogeneous systems with deferred repairs and $K = 16$.

Simulation results are presented comparatively in order to validate the model. For the simulation, the scenario explained in previous sections is considered. The simulation results are within the confidence interval of $\pm 5\%$ of the mean value, with probability of 0.95. The parameters are given as σ jobs/hour, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hr, $K = 8$, $H = 4$. The results are illustrated in figure 6.6.

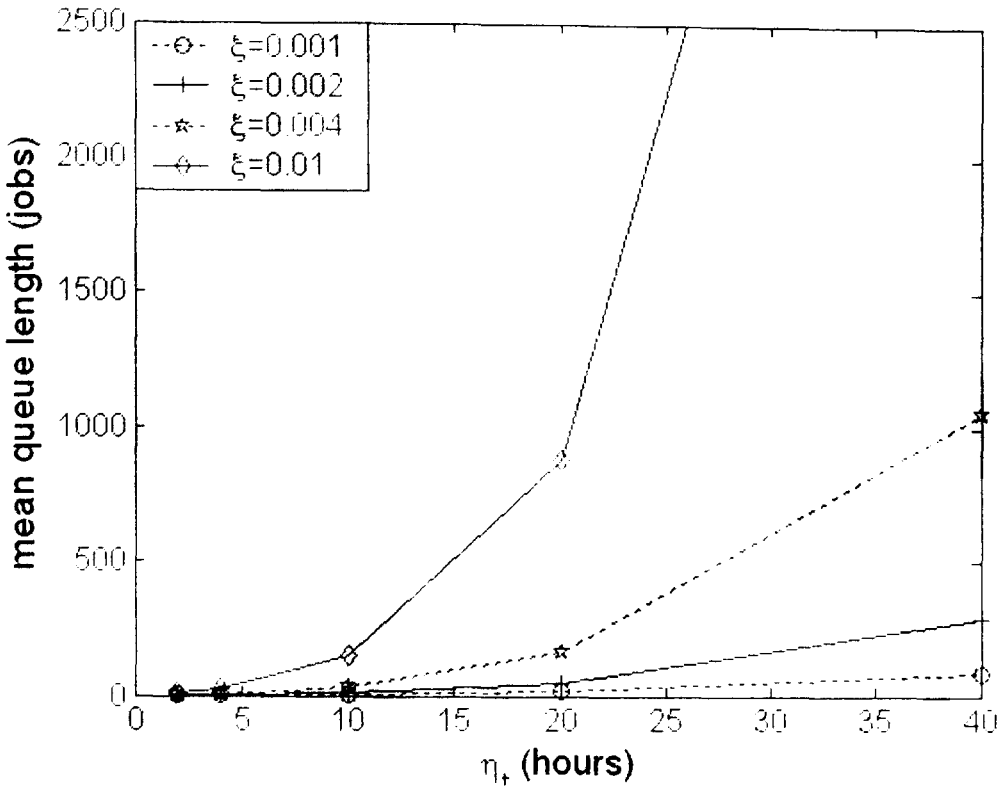

Figure 6.5: MQL as a function of η_t for homogeneous systems with deferred repairs and $K = 4$.

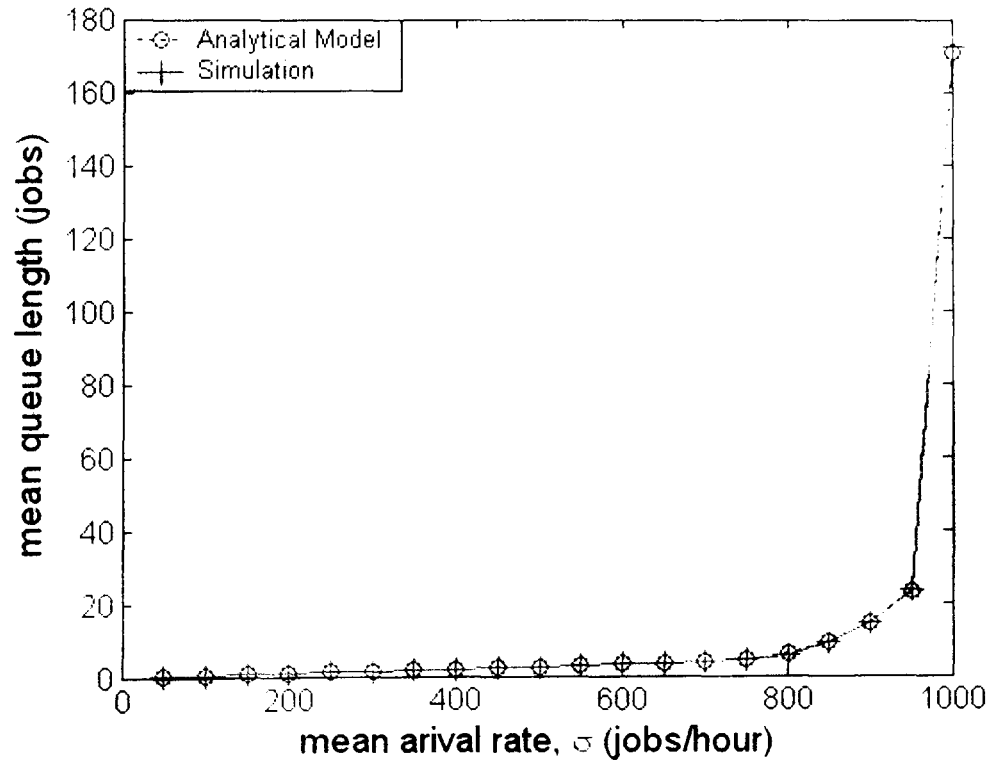

Figure 6.6: Results from simulation and analytical model for homogeneous multi-server systems with deferred repairs.

The final two figures are presented for systems with finite queuing capacities. In figures 6.7 and 6.8 the MQL and PJL values are presented respectively. The other parameters are, σ jobs/hour, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hr, $K = 4$, and $L = 1000$. The results in figure 6.7 show that deferring repairs while there are more than two servers available, is affordable until the σ value reaches 400 jobs/hour. Similarly in figure 6.8 the blocking probability increases after $\sigma = 400$ when the system with $H = 2$ is considered.

The model presented for homogeneous systems with deferred repairs can be used for optimisation of the threshold value H for maintaining an acceptable level of performability. The results obtained for systems with finite, and infinite queuing capacities show that using deferred repair strategies can be affordable. It is possible to include possible reconfiguration and rebooting delays to such a model at the cost of additional states. Homogeneous systems with deferred repairs, and reconfiguration/rebooting delays are considered in the following section.

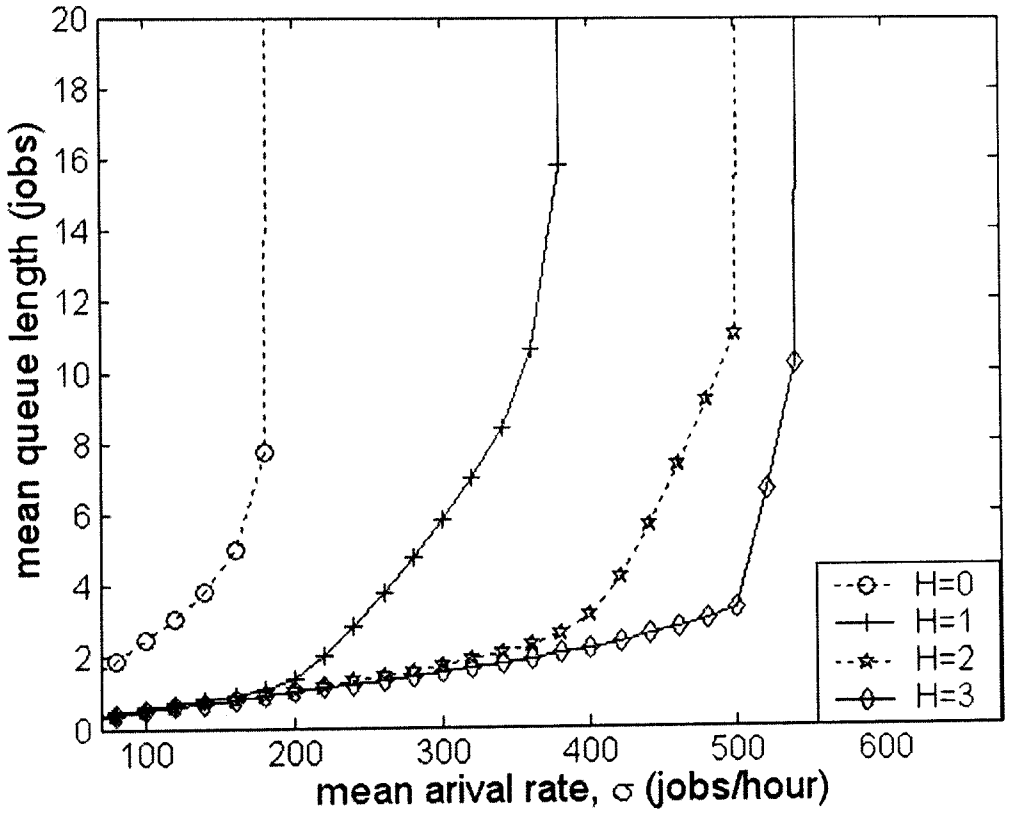

Figure 6.7: MQL as a function of σ for homogeneous systems with deferred repairs $K = 4$, and L=1000.

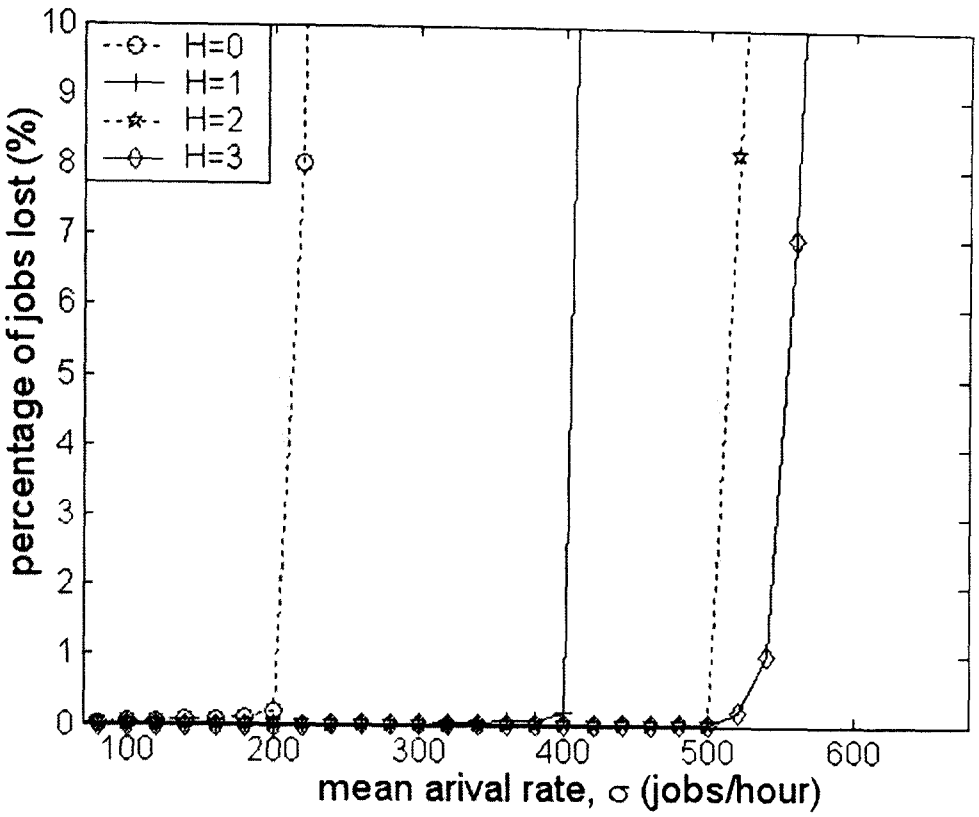

Figure 6.8: PJL as a function of σ for homogeneous systems with deferred repairs $K = 4$, and L=1000.

6.2.3 Modelling Homogeneous Multi-Server Systems with Deferred Repair Strategies and Reconfiguration/Rebooting Delays

Multi-server systems considered are prone to breakdowns. Even if cover is provided with probability c, there would be reconfiguration/rebooting delays to resume the operation in a degraded mode. The systems considered in this section is similar to the systems considered in section 3.1. However, deferred repair strategies are also added to the model. It is also assumed that in case of server failures, if a repairman is present it is possible to continue serving in a degraded mode after a brief reconfiguration delay since the cover facility is present. On the other hand if no repairman is present a longer rebooting action is required. The system under study is similar to the system in figure 6.1 but with rebooting delays when no repairman is not present and reconfiguration delays when a repairman is present.

The mean arrival, service, failure and repair rates of the system are given as σ, μ, ξ, and η respectively. The transportation time is given as $1/\eta_t$. In case of failures if no repairman is present, the system is rebooted with a delay $1/\varphi$ to map out the failed processor and resume operation in a degraded mode. If a repairman is present, then the service is resumed after a shorter reconfiguration delay given as $1/\delta$. Operative states of this system are given in figure

6.9.

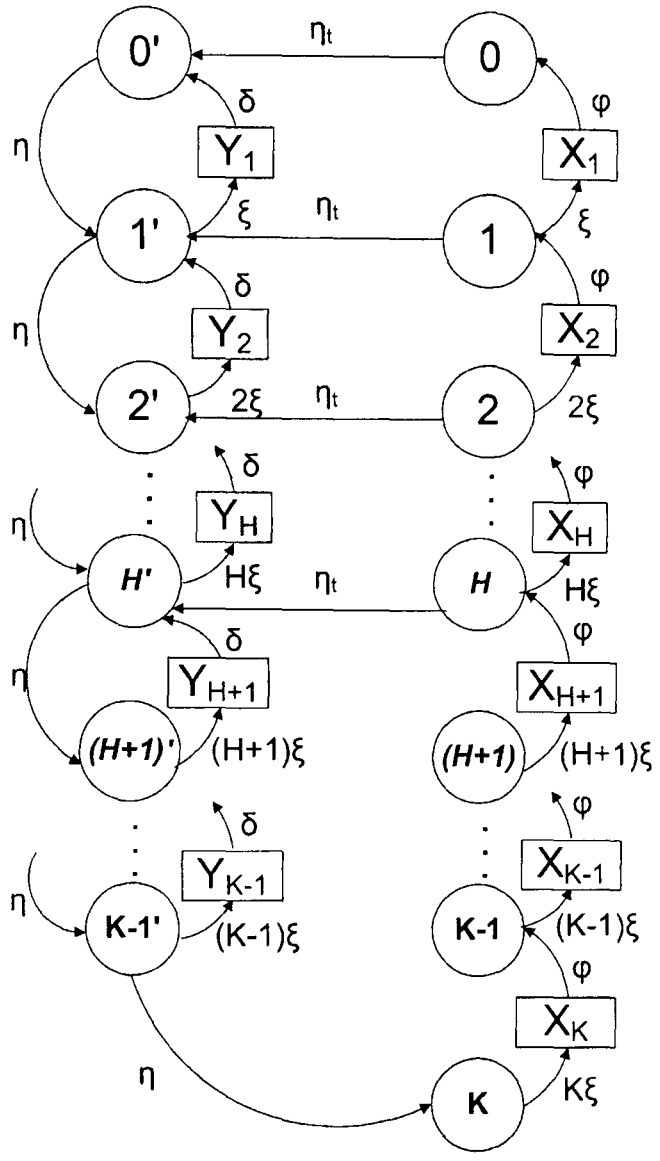

Figure 6.9: Operative states of a homogeneous multi-server system with deferred repairs and reconfiguration/rebooting delays.

In the model given, if a repairman is present, he repairs all broken servers until the system is fully operational. The states labelled 1, 2, ..., H, $(H+1)$, ..., $K-1$, K are the K working states of the multi-server system. In state 0 there are no operative servers and a repairman is present. The states labelled $0'$, $1'$, $2'$, ..., H', $(H+1)'$, ..., $K-1'$ represent the states where the repair facility is present. Finally the states given as X_1, X_2, ..., X_H, $X_{(H+1)}$, ..., X_{K-1}, X_K and Y_1, Y_2, ..., Y_H, $Y_{(H+1)}$, ..., Y_{K-1} are the rebooting and reconfiguration delay states respectively. The total number of states is independent of H and given as $4K$.

The numbering for the transition matrices is done as follows. First $K+1$ numbers $(0, 1,\ldots,K)$ are assigned to states $0, 1, 2, \ldots, H, (H+1), \ldots, K-1, K$, the following K numbers $(K+1, K+2, \ldots, 2K)$ represent states $0', 1', 2', \ldots, H', (H+1)', \ldots, K-1'$. With respect to rebooting delays, numbers $(2K+1, 2K+2, \ldots, 3K)$ are given to states $X_1, X_2, \ldots, X_H, X_{(H+1)}, \ldots, X_{K-1}, X_K$ and for reconfiguration delays, numbers $(3K+1, 3K+2,\ldots,4K-1)$ are assigned to states $Y_1, Y_2, \ldots, Y_H, Y_{(H+1)}, \ldots, Y_{K-1}$. The transition matrices A, B, and C are given as follows:

$$A_j = A = \begin{pmatrix}
0 & \cdots & 0 & \eta_t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \ddots & 0 & \xi & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \eta_t & 0 & \ddots & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K\xi & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \eta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \ddots & 0 & 0 & 0 & \xi & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \eta & 0 & 0 & 0 & \ddots & 0 \\
0 & 0 & \eta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & (K-1)\xi \\
\varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \ddots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \varphi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \ddots & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

$B_j = B$ for $j = 0, 1, \ldots, L$;

$B = \text{Diag}[\sigma, \sigma, \ldots, \sigma]$ of size $(4K) \times (4K)$

$C_j = C$ for $j \geq K$; the threshold $M = K$

$C = \text{Diag}[0, \mu, 2\mu, \ldots, K\mu, 0, \mu, 2\mu, \ldots, (K-1)\mu, 0, \ldots, 0]$

$C_0 = 0$.
$C_j = \text{Diag}[0, Min\{1, j\}\mu, Min\{2, j\}\mu, \ldots, Min\{K, j\}\mu, 0, Min\{1, j\}\mu, Min\{2, j\}\mu, \ldots, Min\{(K-1), j\}\mu, 0, \ldots, 0]$ for $1 \leq j < K$.

where the last $2K-1$ diagonal elements of C and C_j matrices are always zero since they are used to represent rebooting and reconfiguration delay states.

The following section gives numerical results for homogeneous multi-server systems with deferred repairs and reconfiguration/rebooting delays.

6.2.4 Numerical Results for Homogeneous Systems with Deferred Repairs and Reonfiguration/Rebooting Delays

In order to analyse the effects of deferred repairs together with reconfiguration and rebooting delays, numerical results are presented in this section. Simulation results are obtained for these systems as well. Systems with unbounded queues are considered first.

In figure 6.10 an eight server system is considered for various H values ((a) shows loaded systems, (b) shows systems with relatively lighter loads). Other parameters are given as σ jobs/hour, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hr, $1/\delta = 30$ seconds, and $1/\varphi = 10$ minutes. The results clearly show the points where systems with various H values have close performability measures. Although systems with greater H values have better performability as expected, in case of relatively light traffic deferrals are affordable.

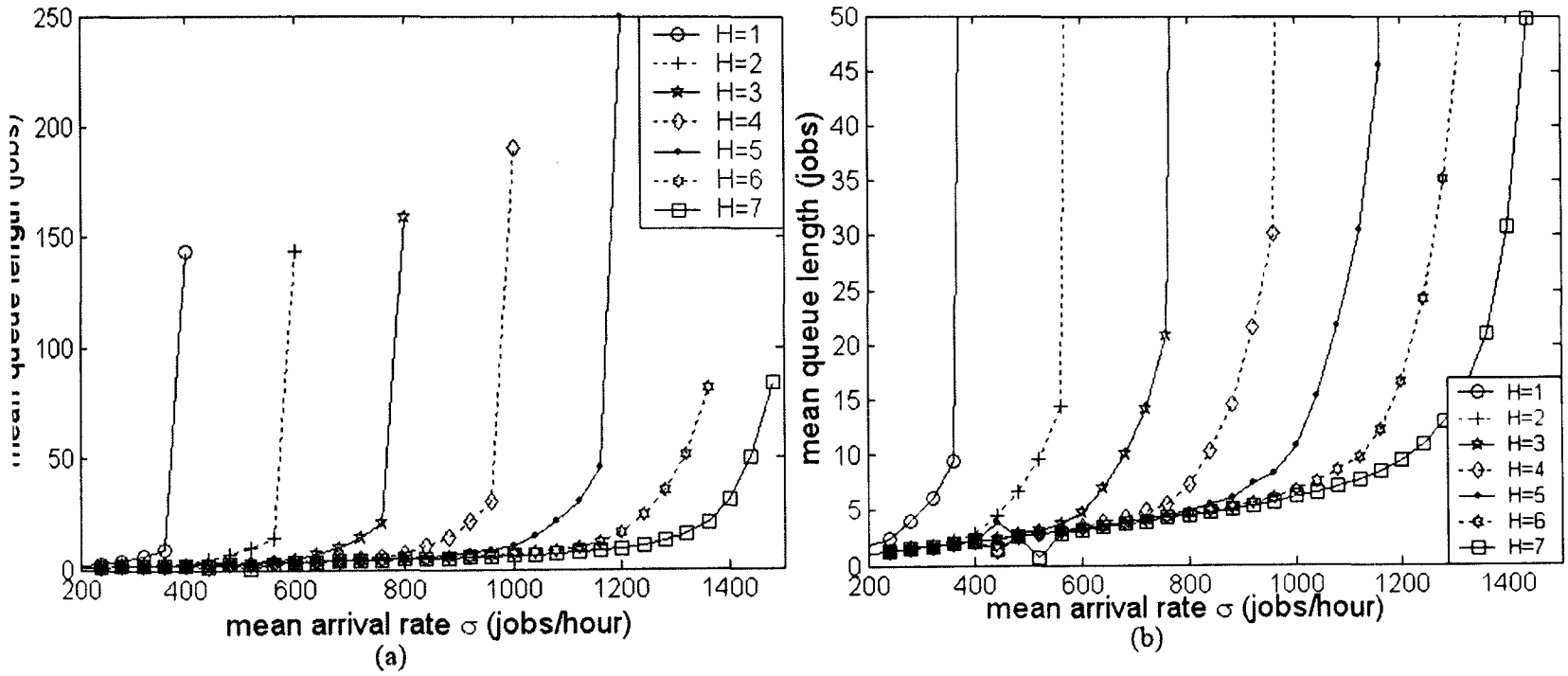

Figure 6.10: MQL as a function of σ for homogeneous systems with deferred repairs, reconfiguration/rebooting delays and $K = 8$.

To show the affects of rebooting delays the results in figure 6.11 are computed. The parameters are given as σ jobs/hour, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hr, $1/\delta = 30$ seconds, and $K = 4$. The results show that effects of rebooting delays increase as the arrival rate increases. Also for systems with smaller threshold values, the effects of rebooting delays on system's performability is more significant.

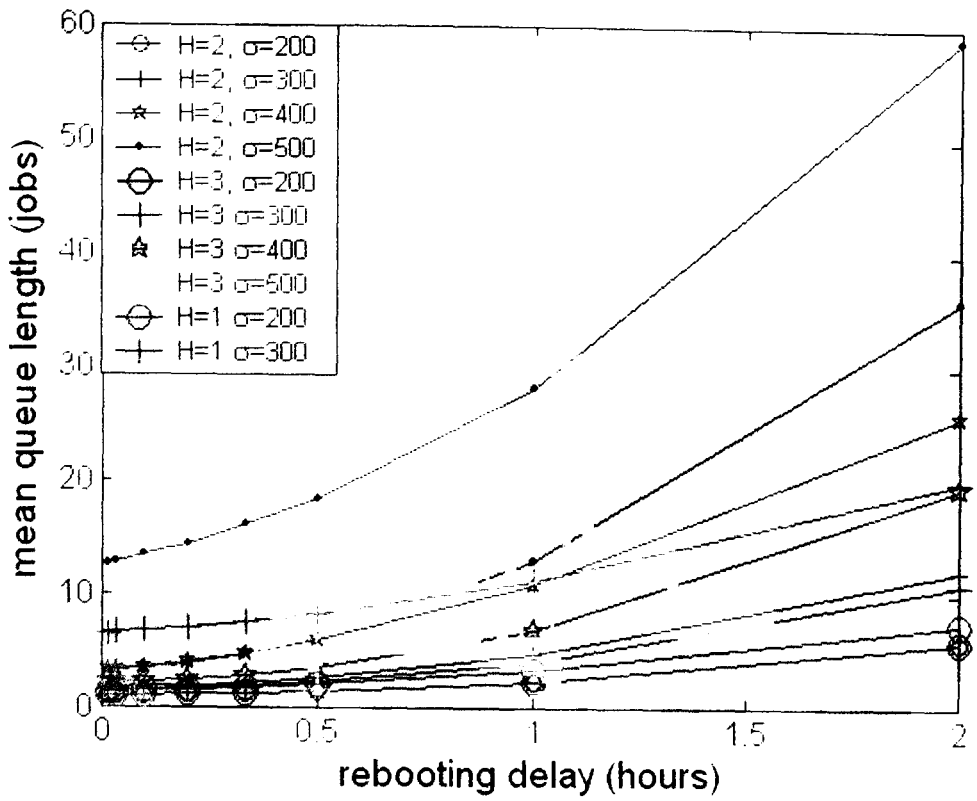

Figure 6.11: MQL as a function of φ, σ and H for homogeneous systems with deferred repairs, and reconfiguration/rebooting delays.

A similar computation is performed to show the effects of reconfiguration delays in figure 6.12. This time MQL is computed as a function of δ. Other parameters are σ jobs/hour, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hr, $1/\varphi = 10$ minutes, and $K = 4$. The results show that reconfiguration delays do not affect the system's performability as much as rebooting delays. This is mainly because of the shorter delay time. Since the servers are down for a shorter duration reconfiguration delays do not affect systems performability even in case of large σ, and small H values.

Simulation results are presented for systems with reconfiguration/rebooting delays as well. For simulation, the scenario explained in this section is considered. The simulation results given are for the process and not for the Markov model developed, and they are within the confidence interval of $\pm 5\%$ of the mean value, with a confidence level of 0.95. The parameters are given as σ jobs/hour, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hr. $\varphi = 2$ minutes, $\delta = 30$ seconds, $K = 4$, $H = 2$. The results are illustrated in figure 6.13.

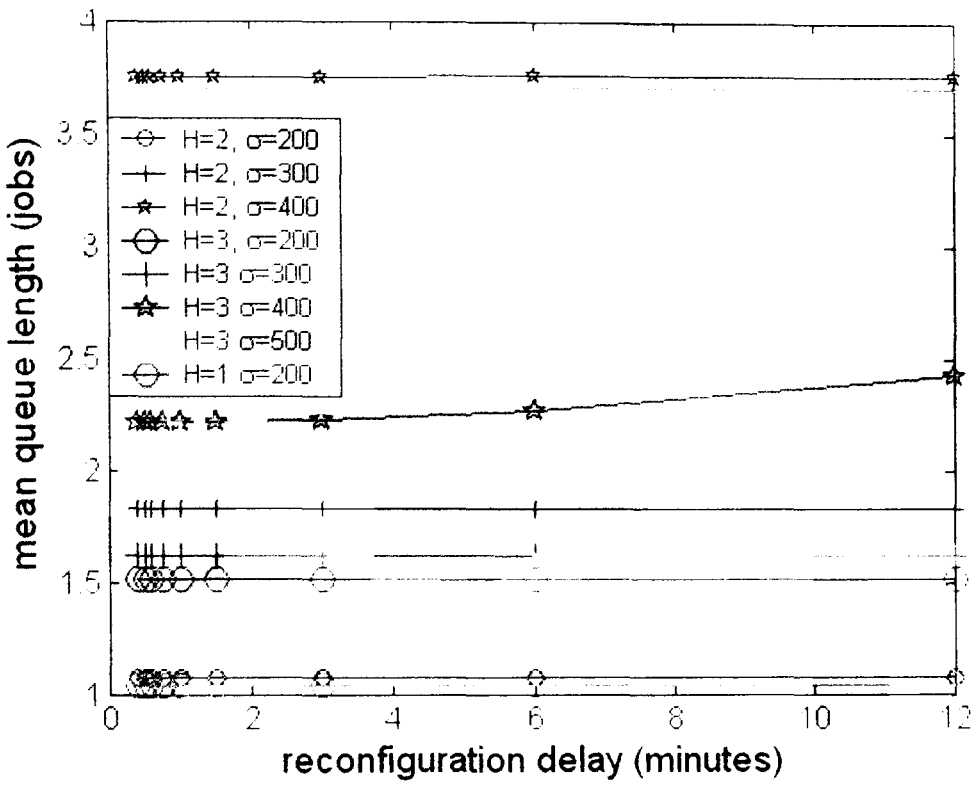

Figure 6.12: MQL as a function of δ, σ and H for homogeneous systems with deferred repairs, and reconfiguration/rebooting delays.

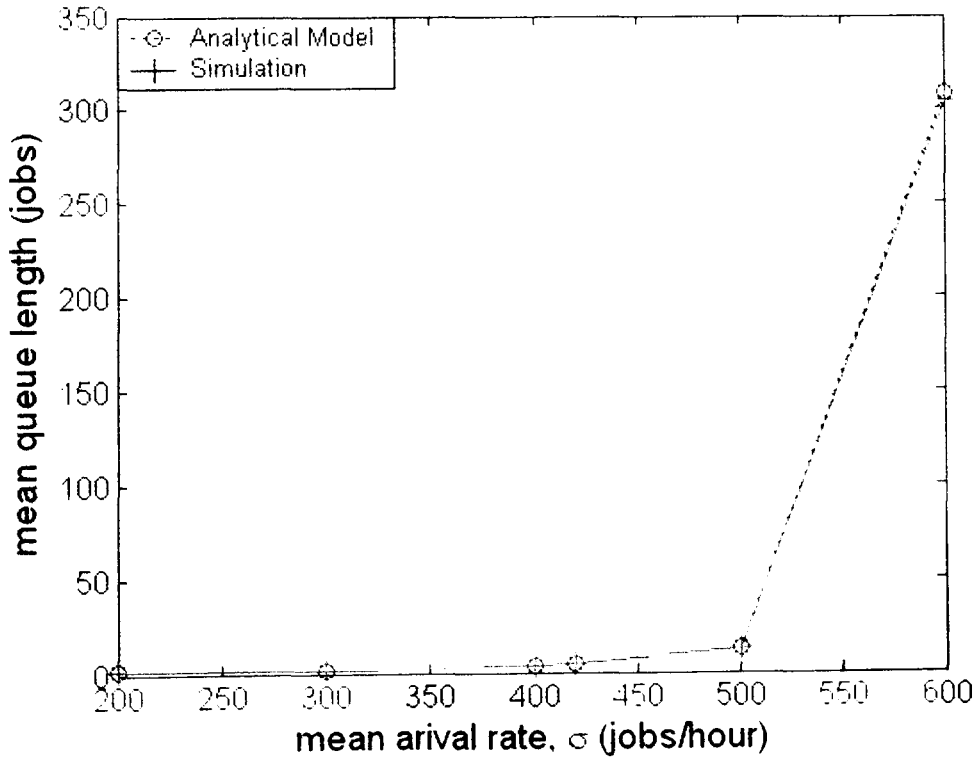

Figure 6.13: Results from simulation and analytical model for homogeneous multi-server systems with deferred repairs, and reconfiguration/rebooting delays.

Finally, results are computed for systems with finite queuing capacities and presented for three and four server systems with respect to σ jobs/hr and for various H values and $L = 1000$. Other parameters are given as $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\eta_t = \eta/2$, $\mu = 200$ jobs/hr, $1/\delta = 1$ minutes, $1/\varphi = 20$ minutes. Computations are performed for both MQL and PJL measures. Figure 6.14 and 6.15 show the MQL and PJL results respectively. The results show that systems with larger numbers of servers and greater H values perform better.

The results obtained for homogeneous multi-server systems with deferred repairs and reconfiguration/rebooting delays show that, even though delays are considered for the system in case of server failures, it is still possible to define a threshold value to defer repairs. Rebooting delays have greater impact on the system's performability. Since in reconfiguration/rebooting period none of the servers are operative, the arriving jobs accumulate in the queue. After the delay period the system continues handling the jobs in a degraded mode (stays in degraded mode if H is small). Because of this the effects of rebooting delays increase as the threshold value decreases and average arrival rates increase.

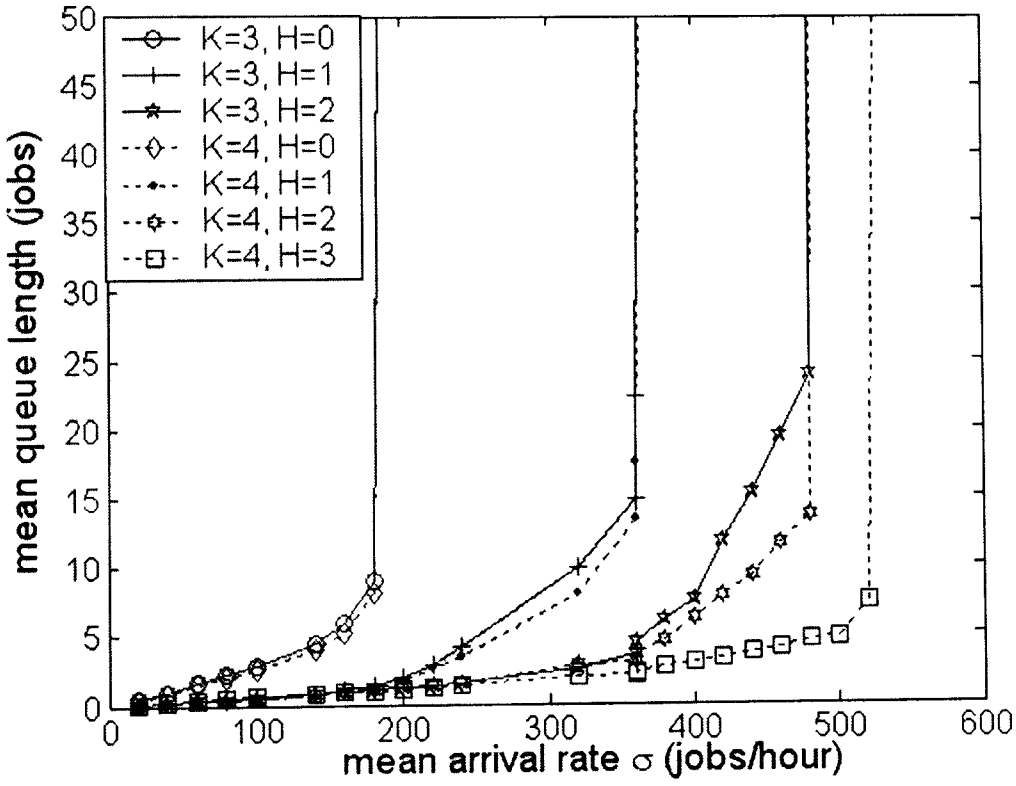

Figure 6.14: MQL as a function of σ for homogeneous systems with deferred repairs, reconfiguration/rebooting delays and L=1000.

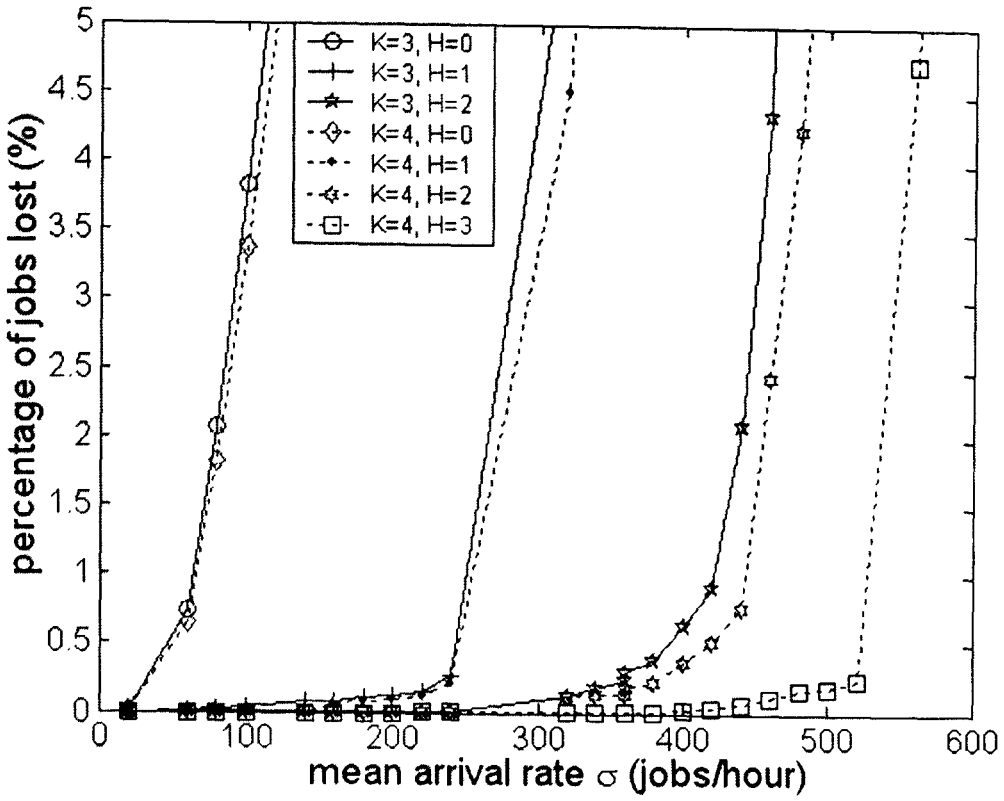

Figure 6.15: PJL as a function of σ for homogeneous systems with deferred repairs, reconfiguration/rebooting delays, and L=1000.

6.3 Effects of Deferred Repair Strategies on Highly Available Clusters with One Head and Several Identical Servers

Network clusters are widely used as parallel processing systems. These systems are tightly-connected networks of computers where each of these computers provide service to incoming job requests (Wagner et. al. 1997, Adams and Vos 2002).

Cluster computing can be used as a relatively low-cost form of parallel processing for scientific and other applications that lend themselves to parallel operations. In chapter 4 performability analysis of multi-server systems with one head and several identical nodes are considered. Beowulf clusters where a number of off-the-shelf PCs are used to form a cluster for providing service to incoming job requests (Leangsuksun et. al. 2004, Engelmann and Scott 2005) are chosen for case studies.

A structure based on a head and multiple computing nodes is used for typical high performance clusters (Leangsuksun et. al. 2004). Typical Beowulf cluster systems are one of the well known multi-server systems which uses this architecture (Leangsuksun et. al. 2004 ,

Leangsuksun et. al. 2005, Engelmann and Scott 2005). Because of this single head node setup, clusters are vulnerable, and this severely limits access to healthy identical nodes in case of head node failures. Performability modelling of typical and highly available Beowulf clusters is considered in chapter 4

Since the most pressing issues of today's cluster architectures are availability and serviceability (Trivedi 2002, Leangsuksun et. al. 2005) various techniques are employed to solve the bottleneck problem caused by the single head node architecture and to implement cluster architecture with high-availability. These techniques include hot standby, and cold standby. In section 4.3 performability models are given for systems using both replacement techniques. In this section, the case where the head node is backed-up with a stand-by head node is considered together with a deferred repair strategy.

6.3.1 Modelling Highly Available Clusters with One Head and Several Identical Servers Using Deferred Repairs

Analytical modelling for various high performance multi-processor systems have widely been investigated in literature. Systems with one head, several identical computing servers and a standby server for head node failures have been considered in section 4.3. This architecture is commonly used to provide high availability to high performance multi-server systems with a single point of failure (Engelmann and Scott 2005).

The multiprocessor system shown in figure 6.16 consists of one head (numbered 1), $K-1$ identical parallel processors, (numbered 2, 3, ..., K), and a standby backup processor for the head processor. The common queue can be bounded with a capacity of L ($L \leq K$), or it can be unbounded. Jobs arrive at the system in a Poisson stream at a mean rate of σ (Hacker and Athey 2001, Gyu et. al. 2004), and join the queue. Jobs are homogeneous and the service rates of the identical processors are the same. The job dependencies in such a system is discussed in chapter 4. The head processor generally has the same service rate as that of the identical ones. The proposed model is applicable even if the head node's service rate is different.

The mean failure rate of the original and standby head nodes is given as ξ_h. The mean failure rate of the computing nodes is ξ_c. Jobs are homogeneous and the mean service rates of the identical processors are given as μ_c for each processor. The service rate of the original and standby head processors is given as μ_h per processor. The switching delay $1/\zeta$ (time needed for standby head processor to take over the control in case of original head processor failures and vice versa) relate to the system and not to individual processors.

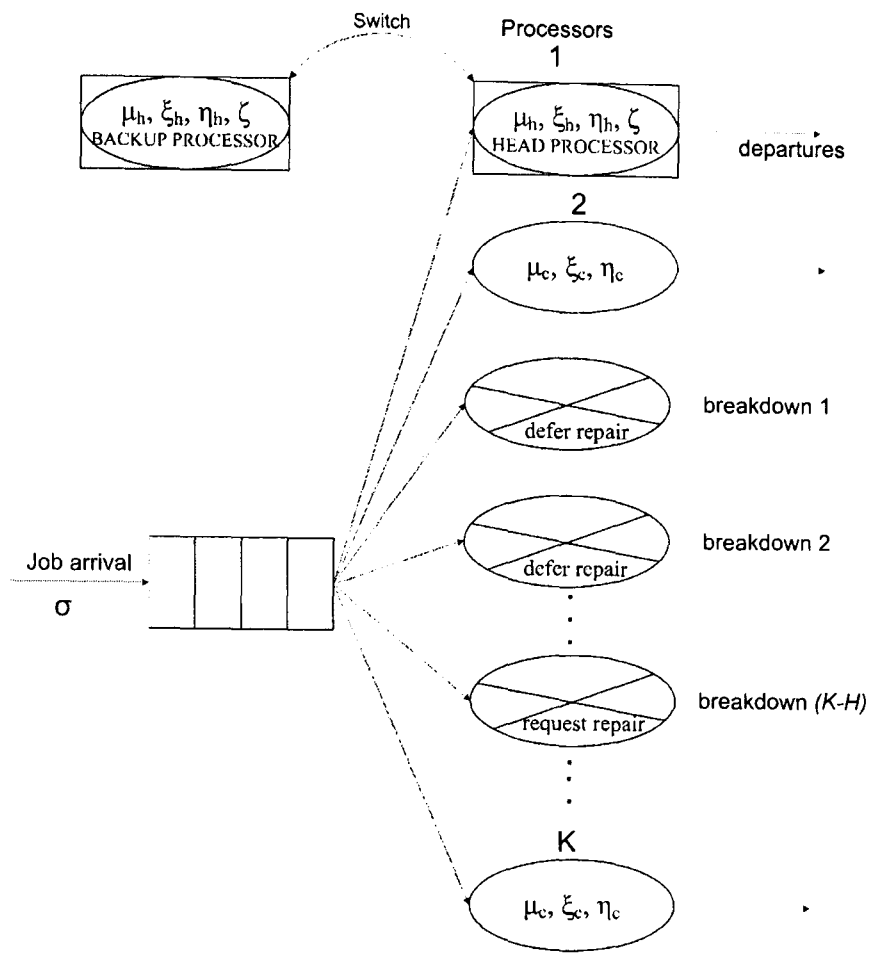

Figure 6.16: A highly available multi-server system with breakdowns and deferred repairs.

At the end of its operative period, a processor breaks down and requires an exponentially distributed repair time. It is assumed that there is a single repair facility. Thus, an inoperative period of a processor would also include the possible waiting time for a repairman. Since the system considered uses deferred repairs strategy, broken identical servers cannot taken to repair until the number of broken identical servers reaches a predefined threshold value $K - H$, where $H < K$. Once the number of broken identical servers becomes $K - H$ a repairman is called. When the head node is broken down repair cannot be deferred and if more than one processor is broken, including the head processor, the repair priority is given to the head node since the head node should be repaired while the standby processor is in use instead of the original head node. This is because the identical processors cannot provide service if the head node and standby node are both non-operative (Leangsuksun et. al. 2004, Leangsuksun et. al. 2005, Engelmann and Scott 2005). If neither the head nor the standby processor is broken, then, the processor to be repaired is chosen randomly amongst the computing processors broken down.

Since transportation is an expensive process in terms of time and labour, it is important to minimise the possible transportation requirements. Because of this, it is reasonable to assume that repairman repairs all servers until the system is fully functional once he/she arrives at the site where the system is located (Temsamani and Carrasco 2002). For this reason, similar to the models presented for homogeneous systems with deferred repairs, the transitions which represent exponentially distributed travel times and repair times are presented independently in this section as well. In cases where a repairman is present all identical servers and the head server have the mean repair times $1/\eta_c$ and $1/\eta_h$ respectively. The mean time required for possible transportation delays is given as $1/\eta_t$ and assumed to follow an exponential distribution. No operative server can be idle if there are jobs awaiting service. All inter-arrival, service, operative and repair time random variables are independent of each other.

If the number of operative identical servers is greater than the number of jobs in the system, then the busy servers are selected randomly. Services that are interrupted by server breakdowns are eventually resumed (perhaps on a different server but at a similar service rate). Failure of the head node does not affect job arrivals and, jobs in the queue remain in the queue without being serviced. In case of systems with bounded queuing capacities the incoming jobs are blocked if the queue is full. Figure 6.17 shows the operative states of the system under study which are used for $I(t)$ in two dimensional representation.

Consider the highly available cluster system with K processors (one head or standby server and K-1 identical servers), given in figure 6.17. μ_h and μ_c are the mean service rates of each of the head node and computing nodes respectively. The mean failure rates for head and computing nodes are given as ξ_h and ξ_c respectively. A single repairman can be called in when the administration decides to do so. The transition rates which are used to represent possible transportation delays are given as η_t. For states where the repair facility is present the mean repair rate is given by η_h and η_c for head (or standby) and identical nodes respectively. It is obvious that the repair facility does not leave the premises (in case there is no permanent repair facility on the premises, e.g. the system is in a remote place) until the whole system is fully operative (all K servers are healthy).

It is a small possibility to have breakdowns before the system is fully repaired but it is not impossible since the servers are put into service once they are repaired. Furthermore, in section 4.3 it has been shown that, assuming that either the head node or the standby processor will be operational at any time, does not change the performance of the system even if there can be short times that both are down. In(Temsamani and Carrasco 2002) a similar assumption is used for reliability-like failure/repair models with exponential failure

and repair time distributions. The repairs are deferred until some condition on the subset of failed components is fulfilled. Then repairman proceeds till the state where all components are healthy is reached, when failure rates are significantly smaller than repair rates. A *rarity* parameter measuring how small failure transition rates are with respect to repair transition rates is defined in (Carrasco 2006) as well.

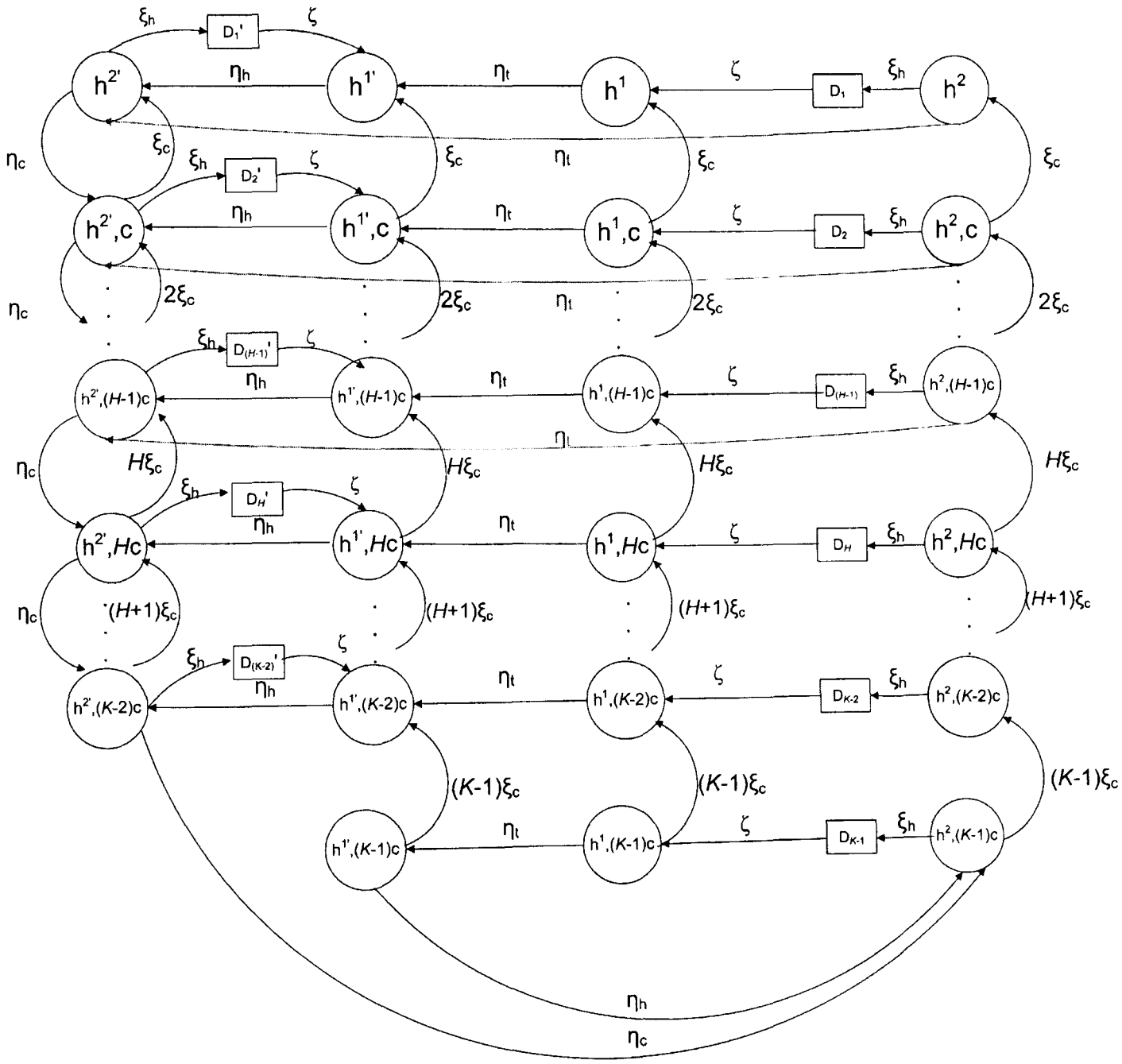

Figure 6.17: Markov Process representing the operative states of the HA system with threshold value H (one head $H-1$ identical processors).

The notation for the states is given as follows:

i. The states labelled (h^2),(h^2, c), ..., $(h^2, (H-1)c)$, (h^2,Hc), ..., $(h^2, (K-2)c)$, $(h^2, (K-1)c)$ are the K states of the multi-server system where both the original and the standby head processors are available. In case of server breakdowns no repairman is present to start repairs immediately. The integer coefficient given before c represents the number of operative identical processors for all states. In case of identical processor breakdowns a repairman is only called if the number of operative identical servers drops to H. On the other hand, in case of head node failures a repairman is called since the head node represents a single point of control.

ii. The states labelled (h^1),(h^1, c), ..., $(h^1, (H-1)c)$, (h^1, Hc), ..., $(h^1, (K-2)c)$, $(h^1, (K-1)c)$ are the K states of the system where the main head processor is broken and the standby head processor takes over the control. No repairman is present but since the head processor is broken, a repairman is called regardless of the number of operative identical processors.

iii. The states $(h^{1'})$, $(h^{1'}, c)$, ..., $(h^{1'}, (H-1)c)$, $(h^{1'}, Hc)$, ..., $(h1', (K-2)c)$, $(h^{1'}, (K-1)c)$ are the K states of the system where the main head processor is broken and the standby head processor takes over the control. In these states a repairman is present. Repair process continues until the system is fully functional. Repair priority is given to the head node. Once the system is fully repaired the repairman leaves the area. The state $(h^2, (K-1)c)$ represents the fully operative system since both head and standby node are operative together with $K-1$ identical processors.

iv. The states $(h^{2'})$, $(h^{2'}, c)$, ..., $(h^{2'}, (H-1)c)$, $(h^{2'},Hc)$, ..., $(h^{2'}, (K-2)c)$ are the $K-1$ states of the multi-server system where both original and the standby head processors are available and the repair man is present. Again the repair process continues until the system is fully functional. In case of head node failures the standby processor takes over the control after a switching delay of $1/\zeta$. In this case the repair priority is again given to the head node. The repairman can leave the area once the system is fully functional.

v. The states labelled as D'_1, D'_2, ..., $D'_{(K-2)}$ and D_1, D_2, ..., $D_{(K-1)}$ are the switching delay states where a repairman is present and no repairman is present respectively.

The total number of states is independent of H and can be expressed as $6K-2$. The system can now be represented by a QBD process with finite or semi-infinite state space as explained in previous sections. The transition matrices A, B and C have the same size of $(6K-2)\times(6K-2)$.

The first K numbers (0, 1, 2, 3, ..., $K-1$) represent the states where both original and the standby head processors are operative and no repairman is present, ((h^2, sc) where s is the number of identical servers). Next K numbers (K, $K+1$, $K+2$, ..., $2K-1$) are assigned to states where the original head processor is broken, the standby head processor takes over the control and no repairman is present (h^1, sc). The next K numbers ($2K$, $2K+1$, $2K+2$, ..., $3K-1$) are assigned for the states where the original head processor is broken, the standby head processor takes over the control and the repairman is present ($h^{1'}$, sc). The next $K-1$ numbers ($3K$, $3K+1$, $3K+2$, ..., $4K-2$) are assigned to the states where both original and standby head processors are operative and repairman is present ($h^{2'}$, sc). The remaining numbers ($4K$, $4K+1$, ..., $6K-3$) are used to represent switching states.

The matrices B and C are given as:

$B_j = B$ for $j = 0, 1, \ldots, L$;

$B = \text{Diag}[\sigma, \sigma, \ldots, \sigma]$ of size $(6K-2)\times(6K-2)$

$C_j = C$ for $j \geq K$; the threshold $M = K$

$C = \text{Diag}[\mu_h, (\mu_h+\mu_c), (\mu_h+2\mu_c), \ldots, (\mu_h+(K-1)\mu_c), \mu_h, (\mu_h+\mu_c), (\mu_h+2\mu_c), \ldots, (\mu_h+(K-1)\mu_c), \mu_h, (\mu_h+\mu_c), (\mu_h+2\mu_c), \ldots, (\mu_h+(K-1)\mu_c), \mu_h, (\mu_h+\mu_c), (\mu_h+2\mu_c), \ldots, (\mu_h+(K-2)\mu_c), 0, 0, \ldots, 0]$

$C_0 = 0$.

If $\mu_h > \mu_c$, the head node has service priority over the identical ones.

Hence, $C_j = \text{Diag}[(\mu_h + Min\{w_c(1), j-1\}\mu_c),\ (\mu_h + Min\{w_c(2), j-1\}\mu_c),\ \ldots,\ (\mu_h + Min\{w_c(K), j-1\}\mu_c),\ (\mu_h + Min\{w_c(1), j-1\}\mu_c),\ (\mu_h + Min\{w_c(2), j-1\}\mu_c),\ \ldots,\ (\mu_h + Min\{w_c(K), j-1\}\mu_c),\ (\mu_h + Min\{w_c(1), j-1\}\mu_c),\ (\mu_h + Min\{w_c(2), j-1\}\mu_c),\ \ldots,\ (\mu_h + Min\{w_c(K), j-1\}\mu_c),\ (\mu_h + Min\{w_c(1), j-1\}\mu_c),\ (\mu_h + Min\{w_c(2), j-1\}\mu_c),\ \ldots,\ (\mu_h + Min\{w_c(K-1), j-1\}\mu_c),\ 0,\ \ldots,\ 0]$ for $1 \leq j < K$.

If $\mu_h \leq \mu_c$, then, the identical nodes have priority over the head.

Hence, $Cj = \text{Diag}[(Min\{w_c(1), j\}\mu_c + Min\{1, j - Min\{w_c(1), j\}\}\mu_h),\ (Min\{w_c(2), j\}\mu_c + (Min\{1, j - Min\{w_c(2), j\}\}\mu_h),\ \ldots,\ (Min\{w_c(K), j\}\mu_c + Min\{1, j - Min\{w_c(K), j\}\}\mu_h),\ (Min\{w_c(1), j\}\mu_c + Min\{1, j - Min\{w_c(1), j\}\}\mu_h),\ (Min\{w_c(2), j\}\mu_c + Min\{1, j - Min\{w_c(2), j\}\}\mu_h),\ \ldots,\ (Min\{w_c(K), j\}\mu_c + Min\{1, j - Min\{w_c(K), j\}\}\mu_h),\ (Min\{w_c(1), j\}\mu_c + Min\{1, j -$

$Min\{w_c(1), j\}\}\mu_h)$, $(Min\{w_c(2), j\}\mu_c + Min\{1, j - Min\{w_c(2), j\}\}\mu_h)$, ..., $(Min\{w_c(K), j\}\mu_c + Min\{1, j - Min\{w_c(K), j\}\}\mu_h)$, $(Min\{w_c(1), j\}\mu_c + Min\{1, j - Min\{w_c(1), j\}\}\mu_h)$, $(Min\{w_c(2), j\}\mu_c + Min\{1, j - Min\{w_c(2), j\}\}\mu_h)$, ..., $(Min\{w_c(K-1), j\}\mu_c + Min\{1, j - Min\{w_c(K-1), j\}\}\mu_h)$, 0, ..., 0] for $1 \leq j < K$.

where $w_c(i)$ is the number of identical processors available in the operative state i. The last $2K - 1$ diagonal elements of matrices C and C_j are always zero since they are used to represent the switching delay states.

For a 4-server system (total number of states = 22) with $H = 2$, matrix A can be expressed as follows:

$$A_j = A =$$

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \eta_t & 0 & 0 & \xi_h & 0 & 0 & 0 & 0 & 0 & 0 \\
\xi_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \eta_t & 0 & 0 & \xi_h & 0 & 0 & 0 & 0 & 0 \\
0 & 2\xi_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 & 0 & 0 & 0 \\
0 & 0 & 3\xi_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \eta_t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \xi_c & 0 & 0 & 0 & 0 & \eta_t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 2\xi_c & 0 & 0 & 0 & 0 & \eta_t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 3\xi_c & 0 & 0 & 0 & 0 & \eta_t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \xi_c & 0 & 0 & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\xi_c & 0 & 0 & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \eta_h & 0 & 0 & 0 & 0 & 0 & 0 & 3\xi_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \eta_c & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \xi_c & 0 & \eta_c & 0 & 0 & 0 & 0 & 0 & \xi_h & 0 \\
0 & 0 & 0 & \eta_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\xi_c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \xi_h \\
0 & 0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

To show the effectiveness of the method presented and evaluate the performance of some typical highly available systems with deferred repairs, numerical results are presented.

6.3.2 Numerical Results for Highly Available, Balanced Fault-Tolerant, Farm Paradigm, Multi-server Systems with Deferred Repairs

Numerical results are presented for both bounded and unbounded systems. Unbounded systems are considered first.

In figure 6.18 and figure 6.19, four and eight server systems are considered respectively ((a) shows loaded systems, (b) shows systems with relatively lighter loads). Other parameters are given as σ jobs/hour, $\xi_h = \xi_c = 0.001$/hr, $\eta_h = \eta_c = 0.5$/hr, $\eta_t = \eta_c/2$, $\mu_h = \mu_c = 200$ jobs/hr, and $\zeta = 1$ min. The effects of different threshold values (H) on the mean queue length of the proposed system are shown. Results show that in case of heavy traffic loads the systems with higher H values perform significantly better. However, in case of relatively lighter traffic loads, the deferral of repairs is possible, since the performability measures do not increase significantly. Table 6.1 gives the MQL performance of a four processor system with various H values in more detail. The sudden jump in MQL is clearly shown on this table.

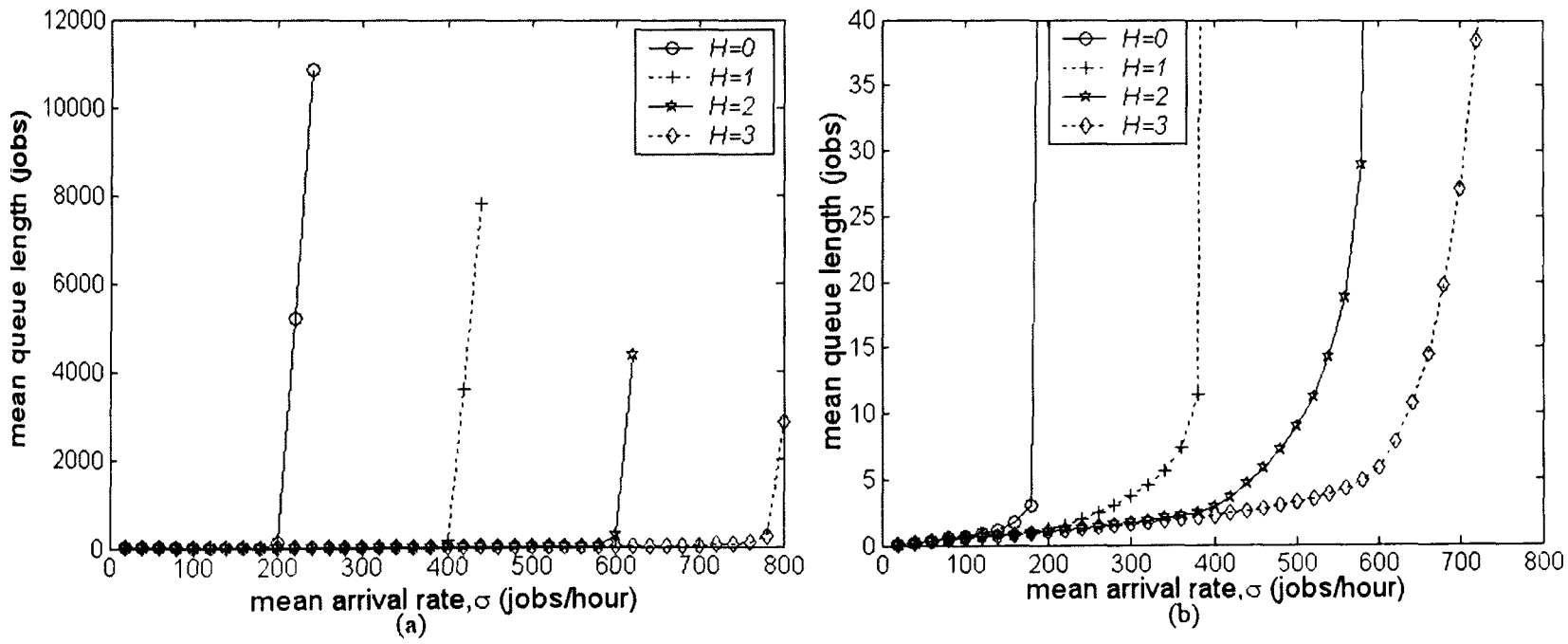

Figure 6.18: MQL as a function of σ and H for HA systems with deferred repairs and $K = 4$.

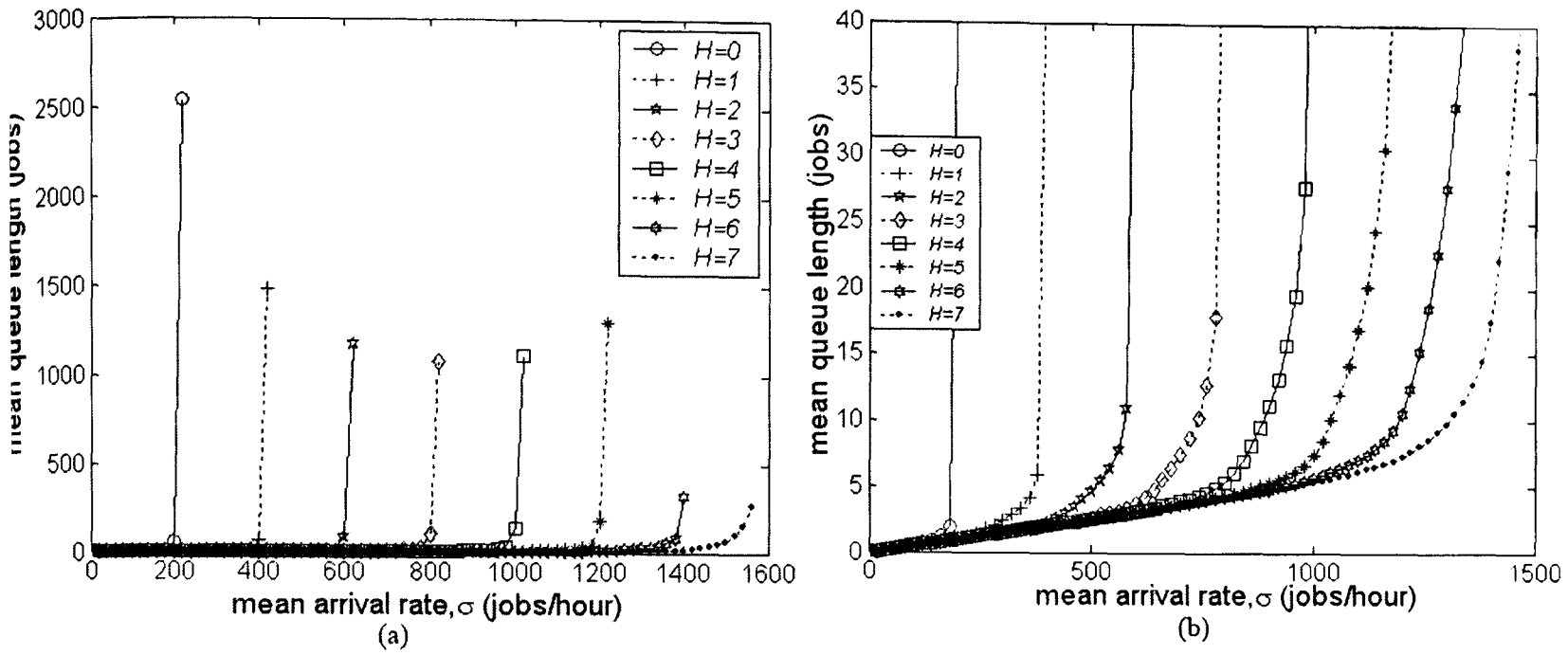

Figure 6.19: MQL as a function of σ and H for HA systems with deferred repairs and $K = 8$.

Table 6.1: Mean queue length versus mean arrival rate for unbounded highly available systems with deferred repairs and $K = 4$.

$\sigma jobs/hour$	$H = 0$	$H = 1$	$H = 2$	$H = 3$
20	0.102	0.1	0.1	0.1
60	0.333	0.302	0.3002	0.3
100	0.634	0.513	0.501	0.5
140	1.135	0.74	0.706	0.701
180	2.988	1.002	0.918	0.904
220	5194.645	1.51	1.144	1.111
260		2.405	1.391	1.325
300		3.65	1.67	1.549
340		5.602	1.999	1.787
380		11.371	2.444	2.049
420		3596.46	3.574	2.357
460			5.778	2.734
500			8.994	3.199
540			14.189	3.813
580			28.995	4.802
620			4382.925	7.787

In figure 6.20, two, four, six and eight server systems are considered. Other parameters are given as $H = K/2$, σ jobs/hour, $\xi_h = \xi_c = 0.001$/hr, $\eta_h = \eta_c = 0.5$/hr, $\eta_t = \eta_c/2$, $\mu_h = \mu_c = 200$ jobs/hr, and $\zeta = 1$ min. Figure 6.20 shows that systems with greater number of servers perform better especially for relatively larger arrival rates.

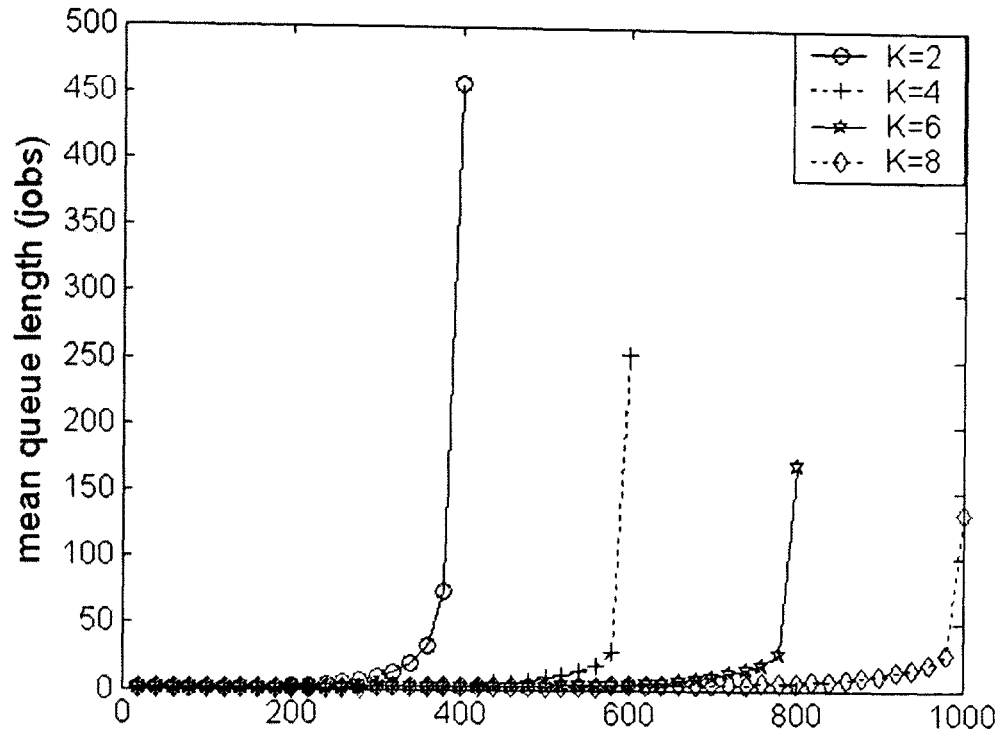

Figure 6.20: MQL as a function of σ and H for HA systems with deferred repairs and various K.

The relationship between the ratio $1/\eta_t$ and MQL is shown in figure 6.21 for various $\xi_h = \xi_c$/hr values, $K = 4$, $\sigma = 400$ jobs/hour, $\mu_h = \mu_c = 200$ jobs/hr, $\eta_h = \eta_c = 0.5$/hr, $\zeta = 1$ min and $H = 2$. Figure 6.21 shows the effects of transportation delay on systems performance. As the mean failure rates of the servers increase the transportation delays affect the system more significantly.

For highly available clusters, the duration of switching delay to switch from a broken down head to the back-up server, depends on system configuration. Hot standby systems take a lot less time to switch while cold standby systems can take a while to do so. Figure 6.22 shows MQL performance as a function of switching delay for various $\xi_h = \xi_c$/hr values where $K = 4$, $\sigma = 400$ jobs/hour, $\mu_h = \mu_c = 200$ jobs/hr, $\eta_h = \eta_c = 0.5$/hr, $\eta_t = \eta_c/2$ and $H = 2$. Figure 6.22 shows that switching delays affect the system performance more significantly, when the failure rates are relatively higher. However the switching delays do not affect the system performance as much as transportation delays since they take shorter time.

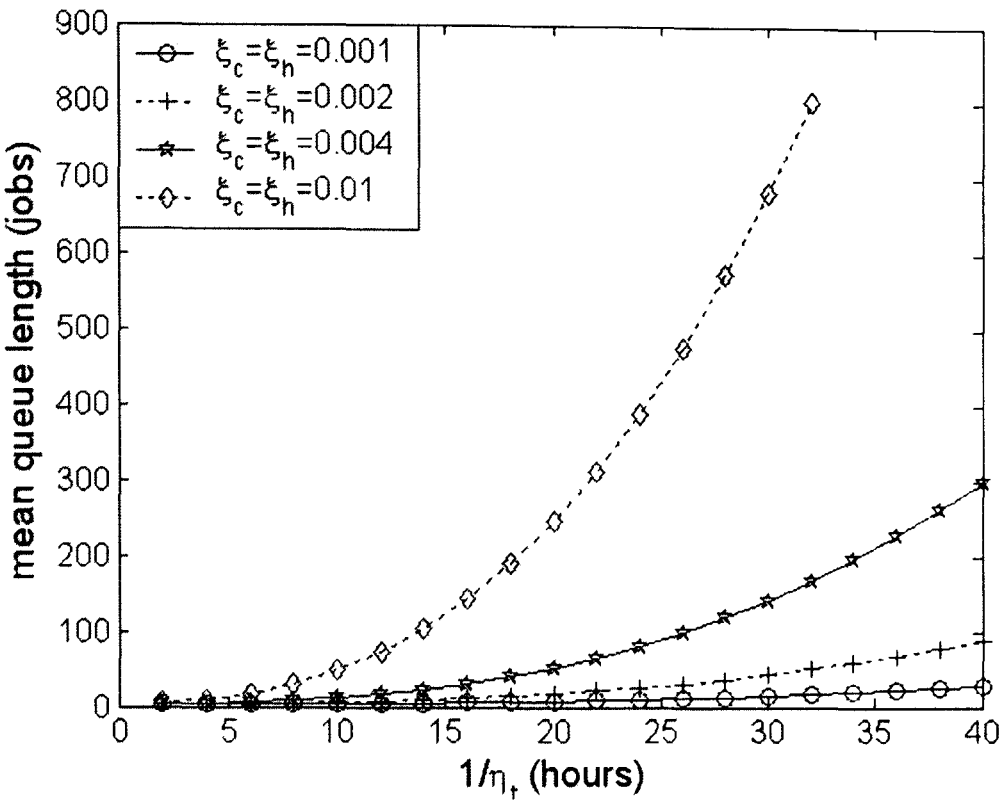

Figure 6.21: MQL as a function of $1/\eta_t$ for HA systems with deferred repairs and $K = 4$.

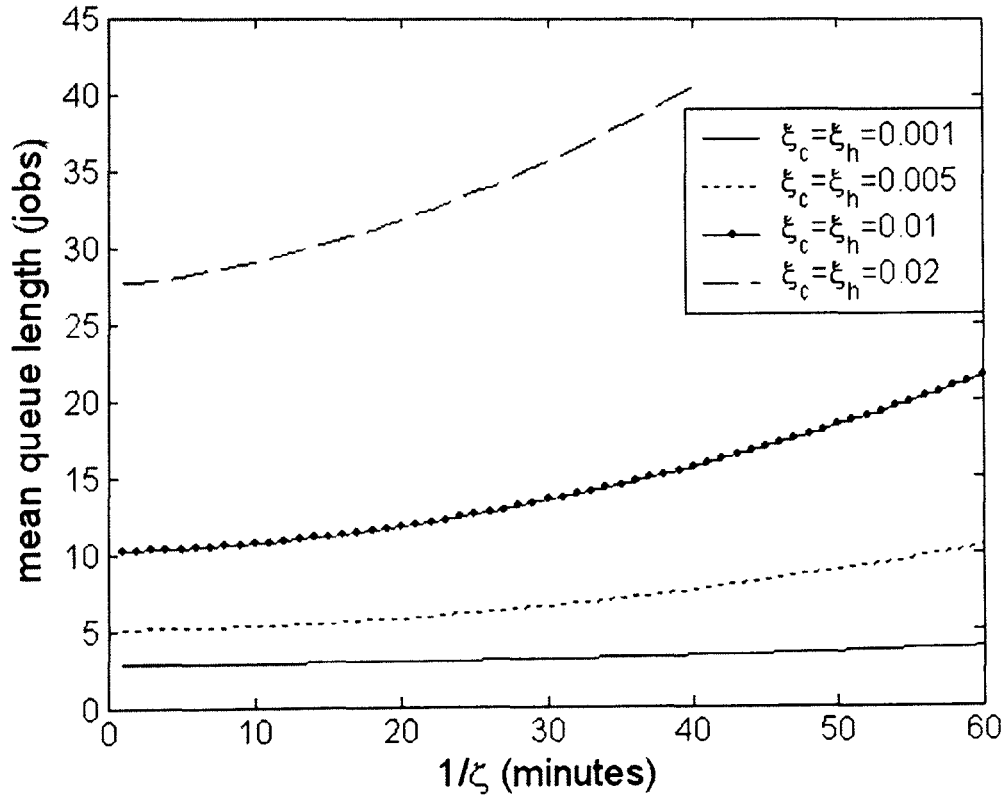

Figure 6.22: MQL as a function of $1/\zeta$ for HA systems with deferred repairs and $K = 4$.

Finally, for an unbounded system, a comparison with highly available farm paradigm systems without deferred repairs is presented. Such a comparison is possible when the repairs are not deferred ($H = K - 1$) and transportation delays are ignored (η_t is taken very high). Comparison is performed between results of the model presented in this section and the model in section 4.3 with no rebooting/reconfiguration delays. Parameters are taken as $K = 4$, σ jobs/hour, $\xi_h = \xi_c = 0.001$/hr, $\eta_h = \eta_c = 0.5$/hr, $\mu_h = \mu_c = 1$ jobs/hr, and $\zeta = 1$ min. Results show good agreement. They are illustrated in figure 6.23.

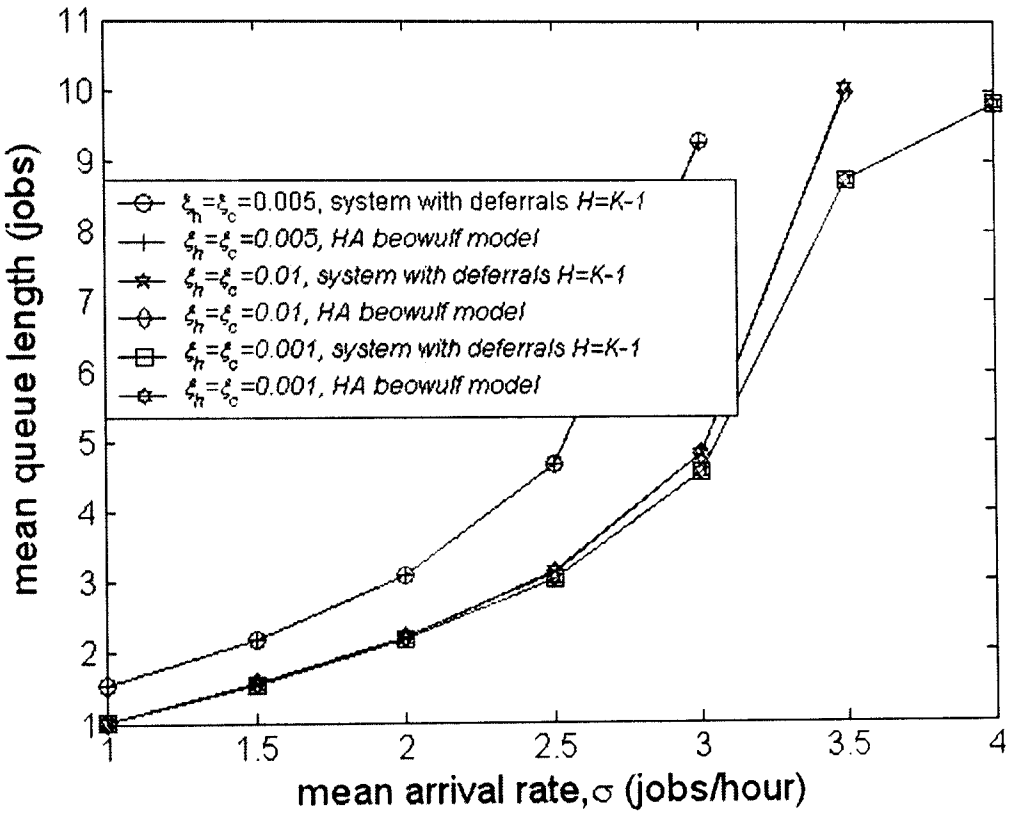

Figure 6.23: Results given for HA systems without deferrals

The results show that when infinite queuing capacities are considered η_t and ζ affect the performability of the systems significantly especially if relatively high failure rates are expected (providing that the condition of being balanced is satisfied). In case of relatively high arrival rates, systems with higher H values perform better. However, if light traffic loads are expected it is possible to defer repairs since the performability measures do not change significantly. Performability of HA systems with deferred repairs and finite queuing capacities are considered in the following parts of this section.

In figure 6.24 a four-server system with bounded queuing capacity is considered. The mean queue length is shown as a function of ζ where $\xi_h = \xi_c$/hr, $\sigma = 400$ jobs/hr, $\mu_h = \mu_c = 200$ jobs/hr, $\eta_h = \eta_c = 0.5$/hr, $\eta_t = \eta_c/2$, $H = 2$ and $L = 1000$. Figure 6.24 shows that when low arrival rates are expected, switching delays affect the systems performance more significantly

for the systems with relatively higher failure rates. Since the incoming traffic is light the queue capacity does not become the main limiting factor.

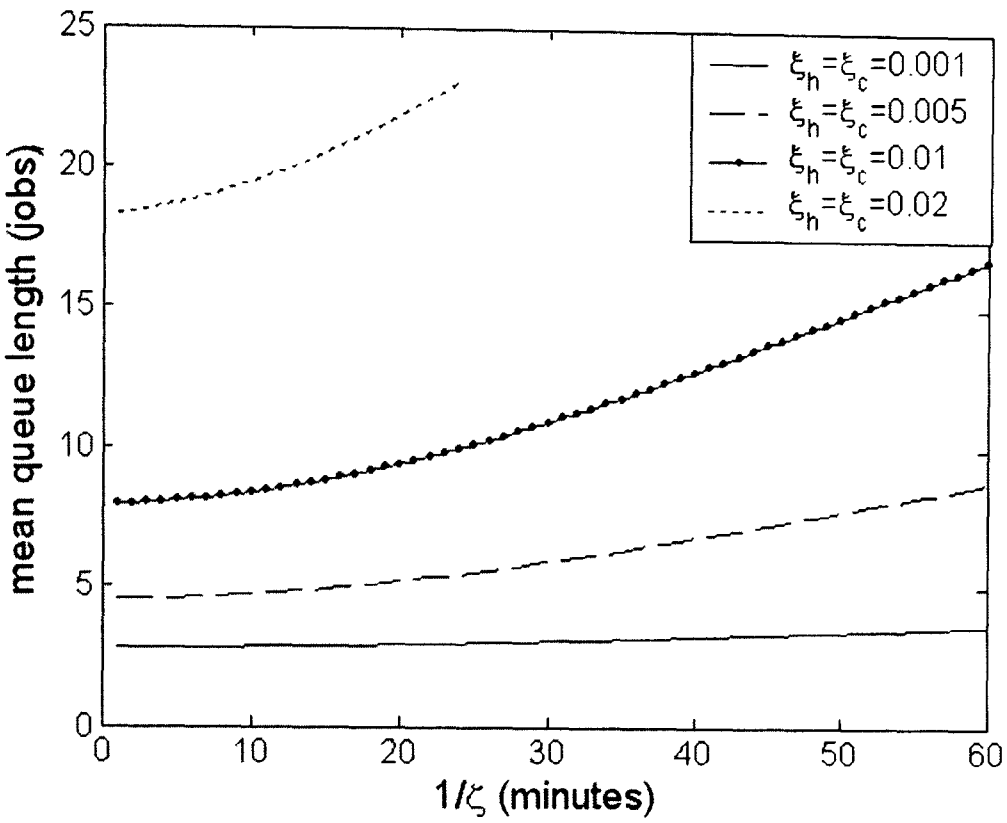

Figure 6.24: MQL as a function of $1/\zeta$ for HA systems with deferred repairs, $K = 4$ and $L = 1000$.

Figure 6.25 shows the mean queue length as a function of η_t where $K = 4$, $\xi_h = \xi_c$/hr, $\sigma = 400$ jobs/hour, $\mu_h = \mu_c = 200$ jobs/hr, $\eta_h = \eta_c = 0.5$/hr, $H = 2$, $1/\zeta = 30$ seconds and $L = 1000$. The effects of transportation delay is more significant for higher $\xi_h = \xi_c$ values. Similar to the previous figure, the queue capacity does not become the limiting factor because of relatively slow arrivals.

In order to investigate the effects of queuing capacity further, the mean queue length and percentage of jobs lost is computed as a function of mean arrival rates for a four server system with various H values. Other parameters are given as σ jobs/hour, $\xi_h = \xi_c = 0.001$/hr, $\eta_h = \eta_c = 0.5$/hr, $\eta_t = \eta_c/2$, $\mu_h = \mu_c = 200$ jobs/hr, $\zeta = 1$ min and $L = 1000$. Figures 6.26and 6.27 show the results of these computations for MQL and PJL respectively.

The results obtained for HA systems with deferred repairs and bounded queues, show that in case of systems with high arrival rates, the queuing capacity becomes the main limiting factor. On the other hand, for systems with lower arrival rates, it is possible to defer the repairs since the performance is not affected significantly.

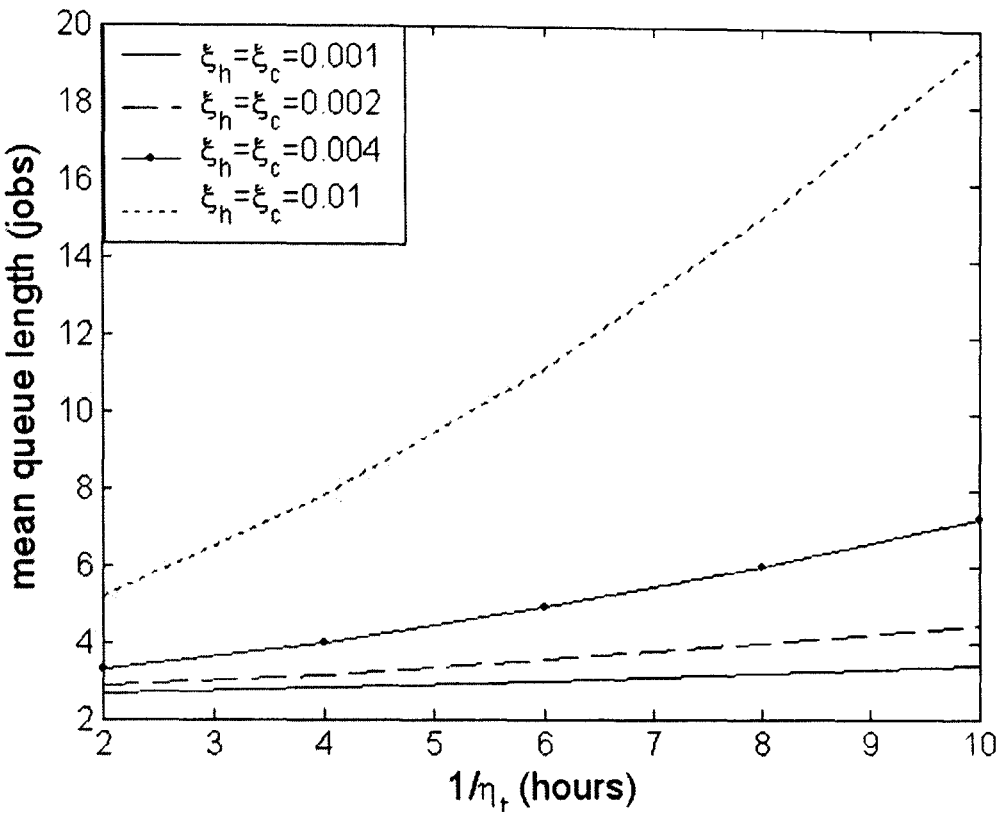

Figure 6.25: MQL as a function of $1/\eta_t$ for HA systems with deferred repairs, $K = 4$ and $L = 1000$.

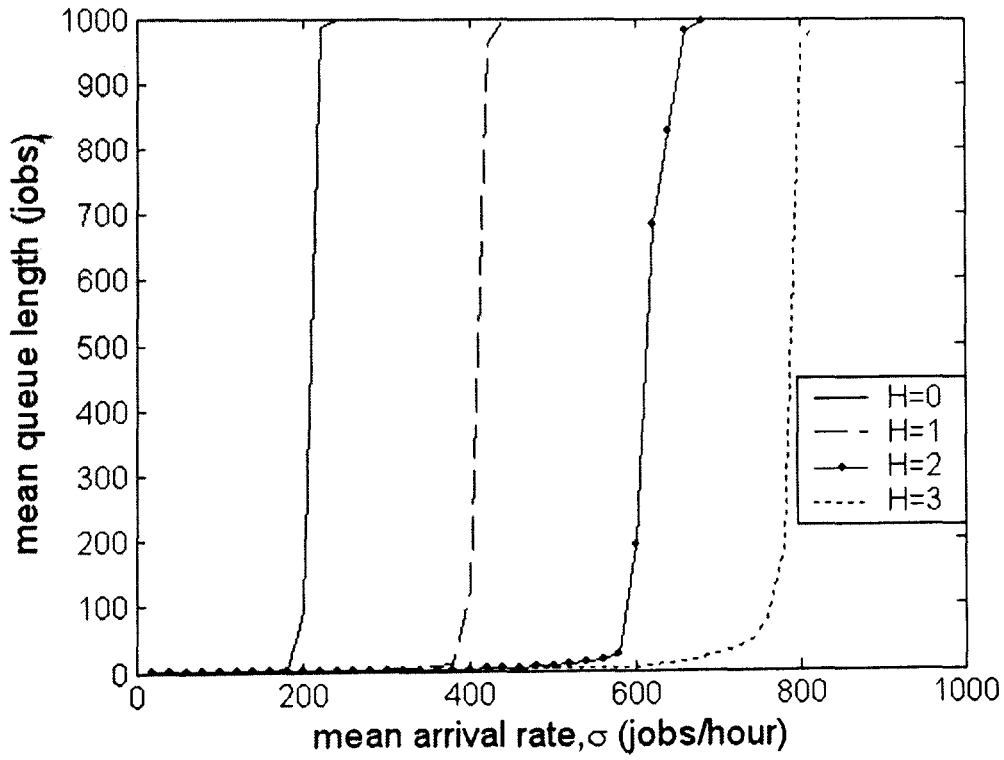

Figure 6.26: MQL as a function of σ for HA systems with deferred repairs, $K = 4$ and $L = 1000$.

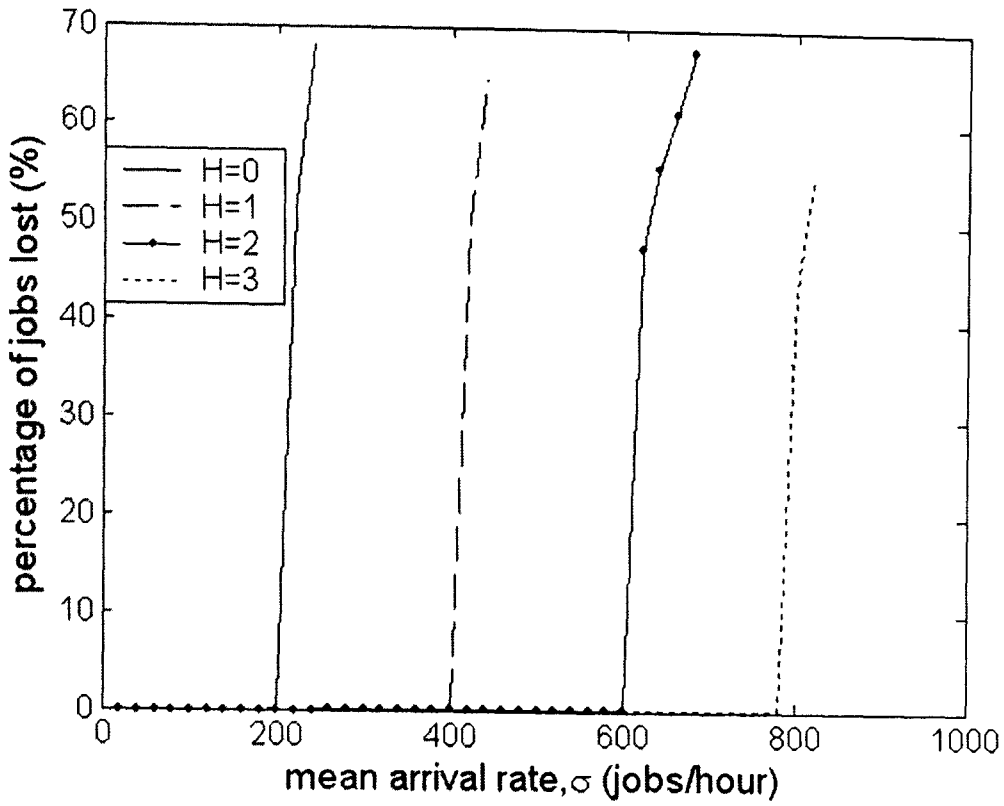

Figure 6.27: PJL as a function of σ for HA systems with deferred repairs, $K = 4$ and $L = 1000$.

6.4 Conclusions and Discussions on Multi-Server Systems with Deferred Repairs Considered

In this chapter homogeneous, and highly available balanced fault-tolerant multi-server systems using farm paradigm are considered.

In the first part of the chapter homogeneous multi-server systems are considered together with possible reconfiguration and rebooting delays. To analyse the effects of various deferred repair strategies, such systems are modelled and solved for exact performability measures for both bounded and unbounded queuing systems. The state probabilities of such systems are derived using the Spectral Expansion method. The queuing issues are considered, investigated and analysed in detail. Numerical results have been obtained and presented for various performability parameters, for both bounded and unbounded systems. Effects of having various threshold values to start deferred repairs, deferred repair rates and reconfiguration/ rebooting delays are analysed. Results show that, the performability of the system depends on the threshold value and deferred repair rate. The effects of deferred repair rate increase as the mean failure rate of the systems increases. The model presented for homogeneous

multi-server systems with deferred repairs, reconfiguration /rebooting delays show that the effects of rebooting delays can be significant but reconfiguration delays does not affect the system significantly. Since the effects of other factors such as the threshold value (H), and transportation delays are greater, the effects of reconfiguration delays are not significant.

Following the sections on homogeneous systems with deferred repairs, highly available clusters with breakdowns, deferred repairs, and switching delays have been modelled for exact solution. The state probabilities in the case of a single head with backup and multiple computing nodes are derived using the Spectral Expansion solution. The queuing issues are considered, investigated and analysed in detail for these systems as well. Numerical results have been obtained and presented for various performability parameters, for both bounded and unbounded systems. Effects of having various threshold values to start deferred repair, various deferred repair rates, and various switching delays are analysed. Switching delays have been chosen so that both hot and cold standby highly available clusters are considered. Results show that, when queue limit is not an important factor, the performability of the system depends on the threshold value and deferred repair rate, and the effects of deferred repair rate increase as the failure rate of the systems increases. Switching delay has also been shown to have a significant impact on system performance. For bounded queuing systems, L is the main factor affecting the mean queue length performance of the system at relatively large σ values. Similar to the homogeneous multi-server systems, HA systems with higher H values perform better in case high arrival rates are expected. Also in case of heavily loaded systems, the queuing capacity becomes the main limiting factor and in case of relatively lower arrival rates, deferrals of the repairs do not affect the performability measures significantly.

The method can be used to compare cost of deferred repairs (in terms of performance degradation) to the cost of traditional repair strategies for various systems. It is possible to invest in additional servers to facilitate differed repairs without experiencing poor performances while cost of operations is reduced due to reduced expenditure on maintaining a repair facility on the premises. The necessary number of additional servers together with savings can be calculated efficiently using the approach presented together with some cost formulae. The models presented are cost effective, extendible and useful for computer and communication systems since most of these systems have queuing considerations.

Chapter 7

An Approach to Ease the State Explosion Problem in Two Stage Tandem Networks

Analytical solutions for two dimensional Markov processes suffer from the state space explosion problem. Especially when both of the random variables ($I(t)$ and $J(t)$) are used to represent the number of jobs in two different queues, the limiting impact of state explosion problem is more significant. Two stage tandem networks are typical examples for such models as considered in (Chakka 1995, Chakka 1998).

In section 2.3.3 existing approaches are analysed for performability evaluation of two stage tandem networks. Because of the limiting effect of random variable $I(t)$, systems with breakdowns and multiple servers at the second stage could not be considered. It is possible to divide the random variable $I(t)$ in order to present the number of jobs in the second stage while the server is operative or broken. However, since $I(t)$ should be divided, the maximum queue capacity that the system can handle will also be divided by total number of operative states of the servers at the second stage. This severely limits the queue capacity that can be handled.

This chapter presents a new approach for analytical modelling of open networks offering improvements in alleviating this problem. The proposed solution is an approximate solution with a high degree of accuracy. Using this approach, two-stage open networks with multiple servers, breakdowns, and repairs at the second stage and feedback can be modelled as three dimensional Markov processes and solved for performability measures. Comparative results show that unlike other solutions such as Spectral Expansion, the new solution approach works for any number of servers.

7.1 Two Stage Tandem Networks Under Study

Markov-modulated queues have a wide range of applications in queuing systems, as shown in previous chapters. Such queues can be described as two dimensional processes where transitions are only possible between adjacent states of a given model. The popularity of two dimensional processes resulted in the development of various numerical procedures for their steady state analysis. These approaches have received considerable attention.

The functional equations arising in the analysis of such processes usually present significant analytic difficulties (Boxma et. al. 1994). These numerical difficulties are frequently caused by large number of steady states. In other words, difficulty caused by the rapid increase in the size of the state space of the underlying Markov process is known as the state space explosion problem. When two dimensional models are considered for two stage tandem networks, the reason of large number of state spaces are the maximum (or infinite) number of jobs in queues of stage 1 or stage 2 ($J(t)$ or $I(t)$). For multi-dimensional models where one component is finite there are good analytic-algorithmic methods, such as Matrix Geometric solution (Neuts 1981), and Spectral Expansion method (Chakka and Mitrani 1992).

These methods can be used to solve the state explosion problem caused by large or infinite number of jobs (El-Rayes et. al. 1999). Although systems where one of the components is infinite can be handled by Spectral Expansion and Matrix Geometric solution methods, as the size of $I(t)$ increases they become computationally expensive. The numerical complexity of the solutions depends on the size of $I(t)$ (Mitrani 2005). That number determines the size of the matrix R used in Matrix Geometric method, and the number of eigenvalues and eigenvectors involved in Spectral Expansion method. In both solution techniques the size of the matrices used depends on the size of random variable $I(t)$. As the sizes of the matrices increase the computational requirements increase significantly. Because of the large size, ill-conditioned (Mitrani 2005) matrices numerical problems occur. Also the numerical stability of the solution gets affected especially when heavily loaded systems are considered. The Spectral Expansion and Matrix Geometric methods are reviewed in (Haverkort and Ost 1997) as well. In his paper, Haverkort, compares Matrix Geometric and Spectral Expansion approaches. Review works highlight strengths as well as weaknesses of numerical techniques used. An important difficulty associated with these methods is the state space explosion problem which limits the size of $I(t)$ which is used to represent the number of jobs in stage 2 of a tandem network to be considered for performability evaluation. Furthermore, when rebooting, reconfiguration, and switching delays are considered together with breakdowns and repairs, this further increases the severity of the problem. Existing solutions are capable of handling large numbers of jobs only at stage 1.

Since the Spectral Expansion (Chakka 1998) method suffers from state space explosion problem as well, Mitrani further improved Spectral Expansion to obtain approximate solutions for heavily loaded Markov-modulated queues using the dominant eigenvalue (Mitrani 2005) for unbounded queues.

In this chapter, the Spectral Expansion technique is adapted to three dimensional Markov Processes to ease the state space explosion problem. The solution is approximate and iterative. The method produces accurate results for tandem systems with bounded queues.

Figure 7.1 shows the queuing system under consideration. In this system, L_j and L_i represent the queuing capacities of the first and second stages respectively. L_j and L_i are both finite and L_j can be a lot larger than L_i. Jobs arrive at stages 1 and 2 at mean arrival rates of σ_1 and σ_2 respectively, both following a Poisson distribution. The mean service rates are μ_1 and μ_2 per server for stages one and two respectively. The first stage is a single server system. Servers at the second stage are homogeneous with mean breakdown rate ξ, and mean repair rate η. The first stage is assumed to be highly reliable. θ_1 and θ_2 represent the fraction of serviced jobs leaving the system after stage 1 and feed back to first stage respectively. K is the total number of servers at stage 2, k is the number of the operative servers and $I(t)$ represents the number of jobs, at stage 2 at time t. When all servers at stage 2 are broken, or the queue is full, stage 1 stops sending jobs to stage 2.

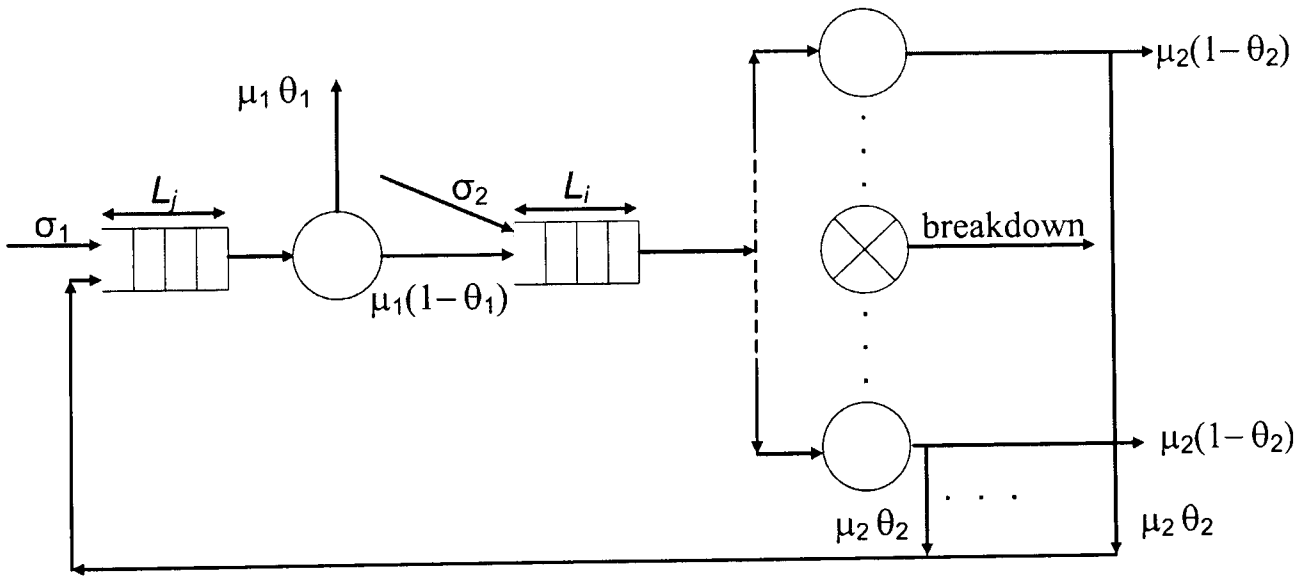

Figure 7.1: A two stage tandem network with multiple servers at the second stage.

The proposed method models the system shown in figure 7.1 as a three dimensional Markov process and solves the resultant model using the Spectral Expansion method together with

an iterative process. Then, the steady state probabilities of the three dimensional Markov chain are calculated. Consider a discrete time, two dimensional Markov process on a finite lattice strip. The Markov Process can be defined as $X = \{I(t), J(t);\ t \geq 0\}$ with a state space of $(\{0, 1, 2, \ldots, L_i\} \times \{0, 1, 2, \ldots, L_j\})$. Then, $i = 0, 1, 2, \ldots, L_i$, and $j = 0, 1, 2, \ldots, L_j$ can be used to represent all possible states, (i, j) on the lattice strip. This system can be solved using existing approaches where most popular ones are explained in (Chakka 1998) and (Ciardo and Smirni 1999). The limiting factor here is the size of I_n, i.e. L_i. When multiple homogeneous servers with breakdowns are considered for stage 2, the size of I_n becomes $(K + 1)(L_i + 1)$, where K is the total number of servers. Clearly, when K is large, only small queuing capacities for stage 2 can be assumed to avoid the ill conditioned matrices caused by large size of transition matrices. In practice however, larger queuing capacities exist.

7.2 The Three Dimensional Solution Approach for Two Stage Tandem Networks with Multiple Servers and Breakdowns at the Second Stage

It is possible to model such networks using three dimensional processes. In this approach I_n becomes independent of K. The Markov process can now be defined as $X_{3D} = \{I(t), J(t), P(t);\ t \geq 0\}$ with a state space of $(\{0, 1, 2, \ldots, L_i\} \times \{0, 1, 2, \ldots, L_j\} \times \{0, 1, 2, \ldots, K\})$. This can be considered as $K + 1$ two dimensional processes, X, let's say X_k, where $k = 0, 1, \ldots, K$.

Each X_k can be considered as an independent two dimensional Markov process. Defining the sum of the steady state probabilities of these two dimensional processes lead to the following sums:

$$S_k = \sum_{i=0}^{L_i} \sum_{j=0}^{L_j} P_{i,j,k} \text{ and } \sum_{k=0}^{K} S_k = 1, \quad k = 0, 1, 2, \ldots, K \tag{7.1}$$

where $P_{i,j,k}$ represent the steady state probabilities.

It is important to calculate S_k for each k. For the system considered on each lattice representing X_k, one step downward transition rates are a function of μ_1, one step upward transition rates are a function of $\mu_2\theta_2$ and σ_1. Lateral transitions are determined by $\mu_1(1 - \theta_1)$, μ_2, and σ_2. Transitions between each lattice X_k occur with rates ξ, and η. The sums (S_k) can be obtained as:

$$S_k = \left[k! \sum_{l=0}^{K} \left(\frac{(\eta/\xi)^{l-k}}{l!} \right) \right]^{-1} \quad k = 0, 1, 2, \ldots, K \tag{7.2}$$

Let's define the matrices u_k for all i, j and k, as $u_k = P_{i,j,k}$, for $0 < j < L_j$, $0 < i < L_i$, $0 < k < K$. For each u_k, $0 < k < K$, the transition matrices are defined in terms of σ_1, σ_2, μ_1, μ_2, θ_1, and θ_2. The possible transitions in each two dimensional process X_k are shown in figure 7.2

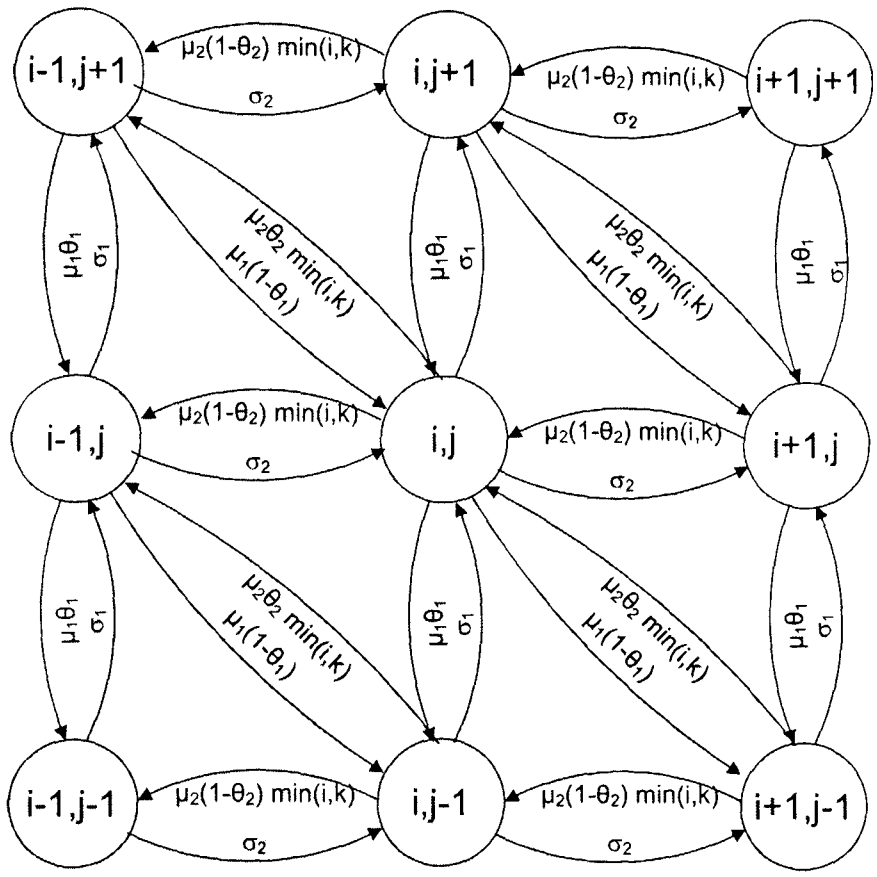

Figure 7.2: The possible transitions within each X_k where $0 < k \leq K$.

When all servers at stage 2 are broken (i.e. $k = 0$) or the queue is full (i.e. $i = L_i$) the terms in transition matrices having $\mu_1(1 - \theta_1)$ are set to zero so that no jobs are sent to stage 2 from stage 1 (it is possible to handle various scenarios). The transition matrices A, A_j, B, B_j, and C, C_j, are used for the Spectral Expansion solution to the processes X_k and can be summarised as follows:

$A_j(i,k)$: Purely lateral transition rates, from state (i,j) to state (l,j), $(i = 0, 1, \ldots, Li;\ l = 0, 1, \ldots, Li;\ i \neq l;\ j = 0, 1, \ldots, L_j)$, caused by a change in the state (i.e., a job has arrived at or departed from the second stage).

$$A_j = A = \begin{pmatrix} 0 & \sigma_2 & 0 & \cdots & 0 & 0 \\ min(i,k)\mu_2(1-\theta_2) & 0 & \sigma_2 & \cdots & 0 & 0 \\ 0 & min(i,k)\mu_2(1-\theta_2) & 0 & \sigma_2 & \cdots & 0 \\ \vdots & 0 & min(i,k)\mu_2(1-\theta_2) & \cdots & \ddots & 0 \\ 0 & \vdots & 0 & \ddots & 0 & \sigma_2 \\ 0 & 0 & \cdots & 0 & min(i,k)\mu_2(1-\theta_2) & 0 \end{pmatrix}$$

$B_j(i,k)$: One-step upward transition rate, from state (i,j) to state $(l,j+1)$, $(i = 0,1,\ldots,L_i;\ \ l = 0,1,\ldots,L_i;\ \$ and $j = 0,1,\ldots,L_j)$, caused by a job arrival at the first stage; external or as a feedback job.

$$B_j = B = \begin{pmatrix} \sigma_1 & 0 & 0 & \cdots & 0 & 0 \\ min(i,k)\mu_2\theta_2 & \sigma_1 & 0 & \cdots & 0 & 0 \\ 0 & min(i,k)\mu_2\theta_2 & \sigma_1 & 0 & \cdots & 0 \\ \vdots & 0 & min(i,k)\mu_2\theta_2 & \sigma_1 & 0 & 0 \\ 0 & \vdots & 0 & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & min(i,k)\mu_2\theta_2 & \sigma_1 \end{pmatrix}$$

$C_j(i,k)$: One-step downward transition rate, from state (i,j) to state $(l,j-1)$, $(i = 0,1,\ldots,Li;\ \ l = 0,1,\ldots,L_i;$ and $j = 0,1,\ldots,L_j)$, caused by the departure of a serviced job from the first stage.

$$C_j = C = \begin{pmatrix} \mu_1\theta_1 & \mu_1(1-\theta_1) & 0 & \cdots & 0 & 0 \\ 0 & \mu_1\theta_1 & \mu_1(1-\theta_1) & \cdots & 0 & 0 \\ 0 & 0 & \mu_1\theta_1 & \mu_1(1-\theta_1) & \cdots & 0 \\ \vdots & 0 & 0 & \mu_1\theta_1 & \ddots & 0 \\ 0 & \vdots & 0 & 0 & \ddots & \mu_1(1-\theta_1) \\ 0 & 0 & \cdots & 0 & 0 & \mu_1\theta_1 \end{pmatrix},$$

for $0 < j \leq L_j$ and $C_0 = [0]$.

From these, the approximate $P_{i,j,k}$ can be obtained using the Spectral Expansion method for each k. Since the steady state probabilities for lattice k are initially calculated independent of lattices $k-1$ and $k+1$, it is important to use a technique to compensate for the unaccounted effects of the latter two lattices on lattice k. This can be achieved through the use of the balance equations given in equations 7.3-7.29. If $s_{x,y}$ represents transitions from lattice x to lattice y where $x \neq y$ and $x, y = 0,1,2,\ldots,K$, then, the transitions can be summarised as in figure 7.3:

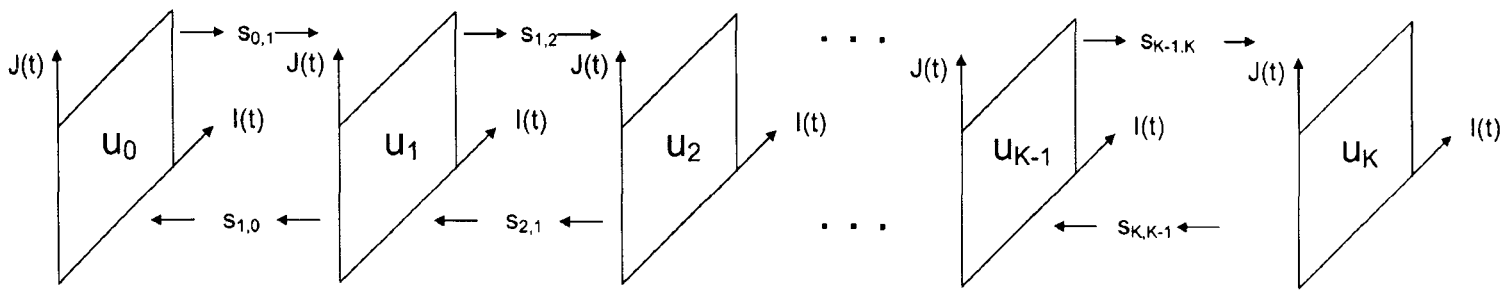

Figure 7.3: The possible transitions between X_ks.

These transitions lead to a set of 27 balance equations in total. All approximate steady state probabilities, u_k, can be calculated for k operative servers, where $k = 0, 1, 2, \ldots, K$ as long as there is a Spectral Expansion solution. The balance equations can be given as follows:

$$P_{0,0,0} = \frac{\xi P_{0,0,1} + \mu_1\theta_1 P_{0,1,0}}{\sigma_1 + \sigma_2 + \eta} \tag{7.3}$$

$$P_{0,j,0} = \frac{\xi P_{0,j,1} + \sigma_1 P_{0,j-1,0} + \mu_1\theta_1 P_{0,j+1,0}}{\sigma_1 + \sigma_2 + \mu_1\theta_1 + \eta}, 0 < j < L_j \tag{7.4}$$

$$P_{0,L_j,0} = \frac{\xi P_{0,L_j,1} + \sigma_1 P_{0,L_j-1,0}}{\sigma_2 + \mu_1\theta_1 + \eta} \tag{7.5}$$

$$P_{i,0,0} = \frac{\xi P_{i,0,1} + \sigma_2 P_{i-1,0,0} + \mu_1\theta_1 P_{i,1,0}}{\sigma_1 + \sigma_2 + \eta}, \ 0 < i < L_i \tag{7.6}$$

$$P_{i,j,0} = \frac{\xi P_{1,j,1} + \sigma_1 P_{i,j-1,0} + \sigma_2 P_{i-1,j,0} + \mu_1\theta_1 P_{i,j+1,0}}{\sigma_1 + \sigma_2 + \mu_1\theta_1 + \eta}, \ 0 < i < L_i, \ 0 < j < L_j \tag{7.7}$$

$$P_{i,L_j,0} = \frac{\xi P_{i,L_j,1} + \sigma_1 P_{i,L_j-1,0} + \sigma_2 P_{i-1,L_j,0}}{\sigma_2 + \mu_1\theta_1 + \eta}, 0 < i < L_i \tag{7.8}$$

$$P_{L_i,L_j,0} = \frac{\xi P_{L_i,Lj,1} + \sigma_1 P_{L_1,L_j-1,0} + \sigma_2 P_{L_i-1,L_j,0}}{\mu_1\theta_1 + \eta} \tag{7.9}$$

$$P_{L_i,0,0} = \frac{\xi P_{L_i,0,1} + \sigma_2 P_{L_i-1,0,0} + \mu_1\theta_1 P_{L_i,1,0}}{\sigma_1 + \eta} \tag{7.10}$$

$$P_{L_i,j,0} = \frac{\xi P_{L_i,j,1} + \sigma_1 P_{L_i,j-1,0} + \sigma_2 P_{L_i-1,j,0} + \mu_1\theta_1 P_{L_i,j+1,0}}{\sigma_1 + \mu_1\theta_1 + \eta} \tag{7.11}$$

$$P_{0,0,k} = \frac{(k+1)\xi P_{0,0,k+1} + \eta P_{0,0,k-1} + \mu_2(1-\theta_2)P_{1,0,k} + \mu_1\theta_1 P_{0,1,k}}{\sigma_1 + \sigma_2 + k\xi + \eta}, \quad 0 < k < K \tag{7.12}$$

$$P_{0,L_j,k} = \frac{(k+1)\xi P_{0,L_j,k+1} + \eta P_{0,L_j,k-1} + \sigma_1 P_{0,L_{j-1},k} + \mu_2\theta_2 P_{1,L_{j-1},k} + \mu_2(1-\theta_2)P_{1,L_j,k}}{\sigma_2 + \mu_1 + k\xi + \eta}, \quad 0 < k < K \tag{7.13}$$

$$P_{0,j,k} = \frac{(k+1)\xi P_{0,j,k+1} + \eta P_{0,j,k-1} + \sigma_1 P_{0,j-1,k} + \mu_2\theta_2 P_{1,j-1,k} + \mu_2(1-\theta_2)P_{1,j,k} + \mu_1\theta_1 P_{0,j+1,k}}{\sigma_1 + \sigma_2 + \mu_1 + k\xi + \eta}, \quad 0 < k < K, \;\; 0 < j < L_j \tag{7.14}$$

$$\begin{aligned} P_{i,0,k} = {} & \frac{(k+1)\xi P_{i,0,k+1} + \eta P_{i,0,k-1} + \sigma_2 P_{i-1,0,k} + min(k,i+1)\mu_2(1-\theta_2)P_{i+1,0,k}}{\sigma_1 + \sigma_2 + \mu_1 + min(k,i)\mu_2 + k\xi + \eta} \\ & + \frac{\mu_1\theta_1 P_{i,1,k} + \mu_1(1-\theta_1)P_{i-1,1,k}}{\sigma_1 + \sigma_2 + \mu_1 + min(k,i)\mu_2 + k\xi + \eta}, \quad 0 < k < K, \;\; 0 < i < L_i \end{aligned} \tag{7.15}$$

$$P_{L_i,0,k} = \frac{(k+1)\xi P_{L_i,0,k+1} + \eta P_{L_i,0,k-1} + \sigma_2 P_{L_i-1,0,k} + \mu_1\theta_1 P_{L_i,1,k} + \mu_1(1-\theta_1)P_{L_i-1,1,k}}{\sigma_1 + min(k,i)\mu_2 + k\xi + \eta}, \quad 0 < k < K \tag{7.16}$$

$$\begin{aligned} P_{i,L_j,k} = {} & \frac{(k+1)\xi P_{i,L_j,k+1} + \eta P_{i,L_j,k-1} + \sigma_1 P_{i,L_j-1,k} + \sigma_2 P_{i-1,L_j,k} + min(k,i+1)\mu_2\theta_2 P_{i+1,L_j-1,k}}{\sigma_2 + \mu_1 + min(k,i)\mu_2(1-\theta_2) + k\xi + \eta} \\ & + \frac{min(k,i+1)\mu_2(1-\theta_2)P_{i+1,L_j,k}}{\sigma_2 + \mu_1 + min(k,i)\mu_2(1-\theta_2) + k\xi + \eta}, \quad 0 < k < K, \;\; 0 < i < L_i \end{aligned} \tag{7.17}$$

$$P_{L_i,L_j,k} = \frac{(k+1)\xi P_{L_i,L_j,k+1} + \eta P_{L_i,L_j,k-1} + \sigma_1 P_{L_i,L_j-1,k} + \sigma_2 P_{L_i-1,L_j,k}}{\mu_1\theta_1 + min(k,L_i)\mu_2 + k\xi + \eta}, \quad 0 < k < K \tag{7.18}$$

$$P_{L_i,j,k} = \frac{(k+1)\xi P_{L_i,j,k+1} + \eta P_{L_i,j,k-1} + \sigma_1 P_{L_i,j-1,k} + \sigma_2 P_{Li-1,j,k} + \mu_1\theta_1 P_{L_i,j+1,k} + \mu_1(1-\theta_1)P_{L_{i-1},j+1,k}}{\sigma_1 + \mu_1\theta_1 + min(k, L_i)\mu_2 + k\xi + \eta},$$

$$0 < k < K,\ \ 0 < j < L_j \tag{7.19}$$

$$\begin{aligned} P_{i,j,k} = [&(k+1)\xi P_{i,j,k+1} + \eta P_{i,j,k-1} + \sigma_1 P_{i,j-1,k} + \sigma_2 P_{i-1,j,k} + min(k, i+1)\mu_2\theta_2 P_{i+1,j-1,k} \\ &+ min(k, i+1)\mu_2(1-\theta_2)P_{i+1,j,k} + \mu_1\theta_1 P_{i,j+1,k} + \mu_1(1-\theta_1)P_{i-1,j+1,k}] \\ &[\sigma_1 + \sigma_2 + \mu_1 + min(k, i)\mu_2 + k\xi + \eta]^{-1}, \end{aligned}$$

$$0 < k < K,\ \ 0 < i < L_i,\ \ 0 < j < L_j \tag{7.20}$$

$$P_{0,0,K} = \frac{\eta P_{0,0,K-1} + \mu_2(1-\theta_2)P_{1,0,K} + \mu_1\theta_1 P_{0,1,K}}{\sigma_1 + \sigma_2 + K\xi} \tag{7.21}$$

$$P_{0,L_j,K} = \frac{\eta P_{0,L_j,K-1} + \sigma_1 P_{0,L_{j-1},K} + \mu_2\theta_2 P_{1,L_{j-1},K} + \mu_2(1-\theta_2)P_{1,L_j,K}}{\sigma_2 + \mu_1 + K\xi} \tag{7.22}$$

$$P_{0,j,K} = \frac{\eta P_{0,j,K-1} + \sigma_1 P_{0,j-1,K} + \mu_2\theta_2 P_{1,j-1,K} + \mu_2(1-\theta_2)P_{1,j,K} + \mu_1\theta_1 P_{0,j+1,K}}{\sigma_1 + \sigma_2 + \mu_1 + K\xi},\ \ 0 < j < L_j \tag{7.23}$$

$$P_{i,0,K} = \frac{\eta P_{i,0,K-1} + \sigma_2 P_{i-1,0,K} + min(K, i+1)\mu_2(1-\theta_2)P_{i+1,0,K} + \mu_1\theta_1 P_{i,1,K} + \mu_1(1-\theta_1)P_{i-1,1,K}}{\sigma_1 + \sigma_2 + min(K, i)\mu_2 + K\xi},$$

$$0 < i < L_i \tag{7.24}$$

$$P_{L_i,0,K} = \frac{\eta P_{L_i,0,K-1} + \sigma_2 P_{L_{i-1},0,K} + \mu_1\theta_1 P_{L_i,1,K} + \mu_1(1-\theta_1)P_{L_{i-1},1,K}}{\sigma_1 + min(K, L_i)\mu_2 + K\xi} \tag{7.25}$$

$$\begin{aligned} P_{i,L_j,K} = &\frac{\eta P_{i,L_j,K-1} + \sigma_1 P_{i,L_{j-1},K} + \sigma_2 P_{i-1,L_j,K}}{\sigma_2 + \mu_1 + min(K, i)\mu_2(1-\theta_2) + K\xi} \\ &+ \frac{min(K, i+1)\mu_2\theta_2 P_{i+1,Lj-1,K} + min(K, i+1)\mu_2(1-\theta_2)P_{i+1,L_j,K}}{\sigma_2 + \mu_1 + min(K, i)\mu_2(1-\theta_2) + K\xi}, \end{aligned} \tag{7.26}$$

$$0 < i < L_i$$

$$P_{L_i,L_j,K} = \frac{\eta P_{L_i,L_j,K-1} + \sigma_1 P_{L_i,L_{j-1},k} + \sigma_2 P_{L_{i-1},L_j,k}}{\mu_1\theta_1 + min(K,L_i)\mu_2(1-\theta_2) + K\xi} \tag{7.27}$$

$$P_{L_i,j,K} = \frac{\eta P_{L_i,j,K-1} + \sigma_1 P_{L_i,j-1,K} + \sigma_2 P_{L_{i-1},j,k} + \mu_1\theta_1 P_{L_i,j+1,K} + \mu_1(1-\theta_1) P_{L_{i-1},j+1,K}}{\sigma_1 + \mu_1\theta_1 + min(K,L_i)\mu_2 + K\xi},$$
$$0 < j < L_j \tag{7.28}$$

$$\begin{aligned} P_{i,j,K} = [&\eta P_{i,j,K-1} + \sigma_1 P_{i,j-1,K} + \sigma_2 P_{i-1,j,K} + min(K,i+1)\mu_2\theta_2 P_{i+1,j-1,K} + \\ &min(K,i+1)\mu_2(1-\theta_2) P_{i+1,j,K} + \mu_1\theta_1 P_{i,j+1,K} + \mu_1(1-\theta_1) P_{i-1,j+1,K}] \\ &[\sigma_1 + \sigma_2 + \mu_1 + min(K,i)\mu_2 + K\xi]^{-1}, \\ &0 < i < L_i,\ \ 0 < j < L_j \end{aligned} \tag{7.29}$$

These balance equations are derived from equation 7.20 by deleting some of its components appropriately (e.g. when $k = K$, all $k+1$ terms are set to zero, for $k = 0$, all terms involving μ_2 and ξ are set to zero etc.). The probabilities obtained within u_k are approximate because transitions between neighbouring lattices are not taken into account. Once the approximate steady state probabilities are calculated, the balance equations ((7.3)-(7.29)) can be used to calculate the steady state probabilities more accurately. An iterative procedure is followed to accurately calculate $P_{i,j,k}$. The procedure can be given as follows:

i. u_k are calculated for $k = 0, 1, 2, \ldots, K$ using the Spectral Expansion method together with equation 7.2,

ii. The balance equations given in 7.3-7.29 are used to calculate the correct steady state probabilities.

iii. Mean queue length is calculated for the queuing system considered.

iv. Second and third steps are repeated until the mean queue length converges sufficiently.

The most recent state probabilities are used to compute various performability measures. The main advantage of the new method is the independent two dimensional solutions for each k value. This makes it possible to handle larger number of servers at the second stage. Also this approach introduces another degree of flexibility to the model. Since each u_k is considered independently, it is possible to handle the transition matrices independently for

each k. For example if the first stage is not sending anything to the second stage in case there are no operative servers at the second stage, it is possible to arrange the transition matrix C in order to handle this scenario for u_0 ($k = 0$ so substitute 0 for $\mu_1(1-\theta_1)$).

7.3 Addressing the Accuracy of the Three Dimensional Solution Approach

In this section the numerical results are presented comparatively with the results obtained from the Spectral Expansion method. Simulation results are also carried out to address the accuracy since the Spectral Expansion method cannot handle large number of servers at the second stage. First, the numerical results obtained from the new approach are compared with the numerical results obtained from the Spectral Expansion method. For the numerical results obtained from the Spectral Expansion method, the random variable $I(t)$ is divided into sections to represent the number of jobs at stage two for various operative states of the servers. Since the number of servers at the second stage limits the Spectral Expansion method, computations are performed up to $K = 2$. The CPU times are also given for comparison where $K = 2$, $L_j = 1000$, $L_i = 15$ (It has not been possible to obtain results using Spectral Expansion and $L_i > 15$ for $K = 2$), $\mu_1 = 5$ jobs/hr, $\mu_2 = 6$ jobs/hr, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\theta_1 = 0.6$, $\theta_2 = 0.2$, and $\sigma_2 = 0$ /hr. The results are illustrated in tables 7.1 and 7.2 for the mean queue lengths of first and second stages respectively (MQL_j, MQL_i).

Table 7.1: MQL for stage one and associated cpu times as a function of the mean arrival rate of stage one, for $K = 2$.

σ_1	MQL_j *Spectral Expansion*	MQL_j $3-D$ *Approach*	*CPU time Spectral Expansion(seconds)*	*CPU time* $3-D$ *Approach (seconds)*
0.5	0.121952	0.121952	69.82	42.1
1.0	0.277779	0.277779	41.289	39.937
1.5	0.483874	0.483874	28.451	40.458
2.0	0.769237	0.769237	17.856	36.332
2.5	1.190488	1.190487	20.309	32.216
3.0	1.875025	1.875014	17.776	24.805
3.5	3.181875	3.181856	20.399	27.369
4.0	6.666817	6.666859	31.895	35.07
4.5	45.006128	45.005791	34.028	306.831
5.0	988.459315	976.44794	19.558	326.93

Table 7.2: MQL for stage two as a function of the mean arrival rate of stage one, for Spectral Expansion and $3-D$ approach for $K=2$.

σ_1	MQL_i *Spectral Expansion*	MQL_i $3-D$ *Approach*
0.5	0.036249	0.036249
1.0	0.07258	0.07258
1.5	0.109066	0.109066
2.0	0.145781	0.145781
2.5	0.1828	0.1828
3.0	0.220202	0.2202
3.5	0.258065	0.258063
4.0	0.296474	0.296472
4.5	0.335517	0.335515
5.0	0.336695	0.336495

A laptop with 1.05 GHz AMD Athlon XP 2200 processor and 704 MB RAM, MS VC++ 6.0, and the NAG library (for Spectral Expansion only) are used to achieve the results in tables 7.1 and 7.2. Results show that the new method is accurate and the error rate is less than %3.38. Clearly, results are obtained at the expense of computation time using the proposed approach. When it is compared with Spectral Expansion method, the new approach gives very close results at a fraction of the computation time. CPU times of the $3-D$ approach increase as the load of the system increases. The Spectral Expansion method is superior for small K. However, as K becomes larger, the state space explosion problem shows itself as a limiting factor. The main advantage of the proposed technique is the ability to work with large K.

A special simulation software is written in C++ language to simulate the considered system. The results obtained from the simulations are compared with the analytical results which are obtained by applying the 3-D solution approach to the Markov model of the system. The simulator measures the mean queue lengths at both first and second stages (MQL_j and MQL_i) for a given set of system parameters. The simulations are continued until a certain confidence interval is achieved. Each result obtained from the simulations is within the confidence interval of $\pm 5\%$ with a probability of 0.95.

In figure 7.4, MQL_j values are presented as a function of mean arrival rate for 4, 8, and 12 servers at the second stage. The other parameters are $L_j = 100$, $L_i = 40$, $\mu_1 = 5$ jobs/hr, $\mu_2 = 0.5$ jobs/hr, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\theta_1 = 0.6$, $\theta_2 = 0.2$, and $\sigma_2 = 0$ jobs/hr. MQL_j results obtained using the new approach are compared with the simulation results. The maximum error rate for performability measure MQL_j is less then %3.88 when analytical

and simulation results are compared.

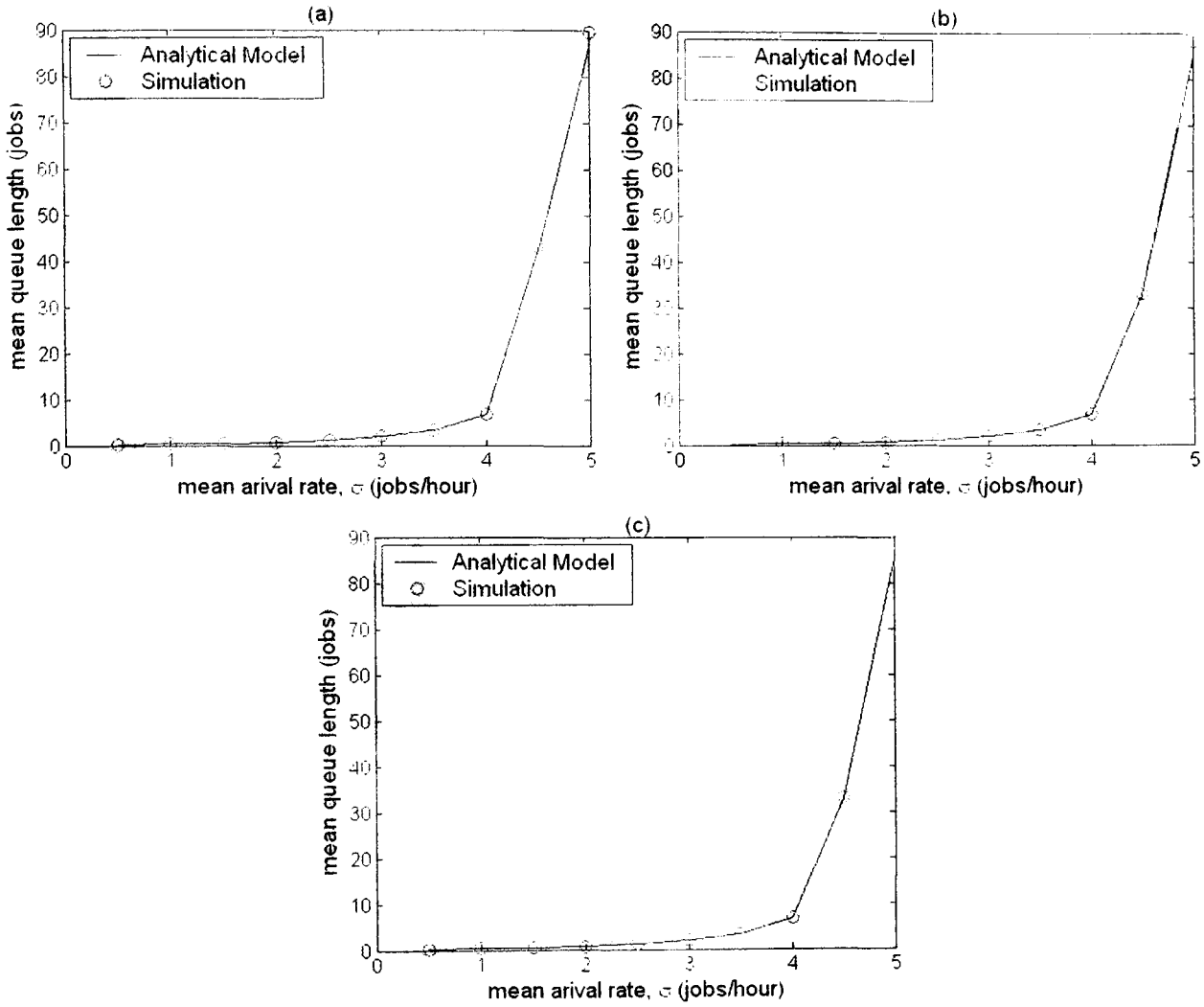

Figure 7.4: MQL_j as a function of σ_1 for tandem networks with multiple servers at the second stage where $L_i = 40$ and $L_j = 100$ ((a) $K = 4$, (b) $K = 8$, (c) $K = 12$).

Figure 7.5 shows the MQL_i values computed with the same parameters used in obtaining figure 7.4. For MQL_i values the error rate is less than %4.31 when the analytical results are compared to the simulation results.

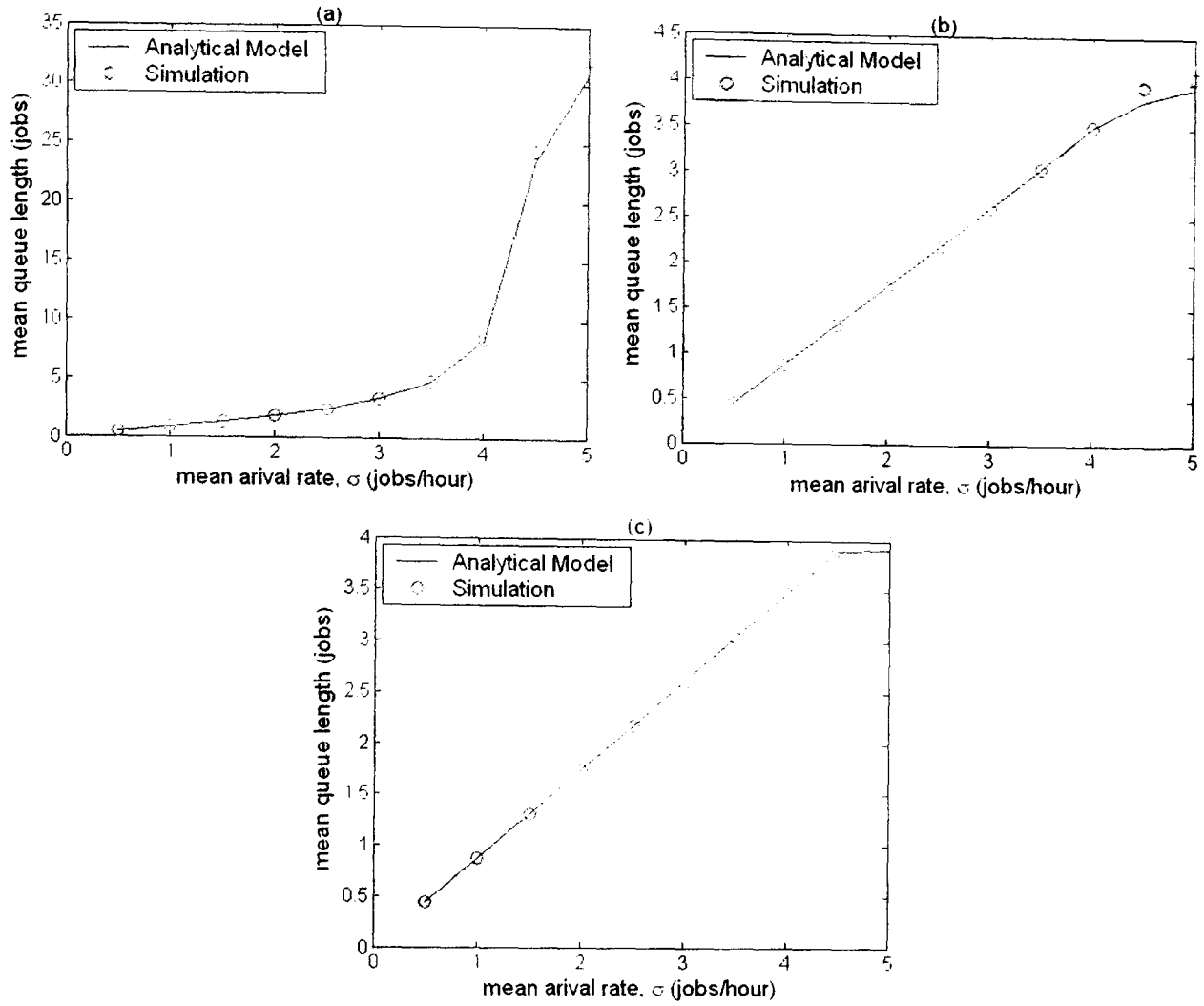

Figure 7.5: MQL_i as a function of σ_1 for tandem networks with multiple servers at the second stage where $L_i = 40$ and $L_j = 100$ ((a) $K = 4$, (b) $K = 8$, (c) $K = 12$).

Same parameters are used for 4 and 8 server systems with external arrivals at the second stage where $\sigma_2 = 3$ and service rate of the second stage $\mu_2 = 1$. The results are shown in figure 7.6. For MQL_j and MQL_i values the error rates are less then %4.42 and %4.66 respectively.

In figures. 7.7 and 7.8 MQL_j and MQL_i measures are presented respectively as a function of mean arrival rate. This time parameters are taken as $L_j = 1000$, $L_i = 15$, $\mu_1 = 5$ jobs/hr, $\mu_2 = 0.5$ jobs/hr, $\xi = 0.001$/hr, $\eta = 0.5$/hr, $\theta_1 = 0.5$, $\theta_2 = 0.5$, and $\sigma_2 = 0$ jobs/hr. Again tandem systems with 4, 8, and 12 servers at the second stage are considered.

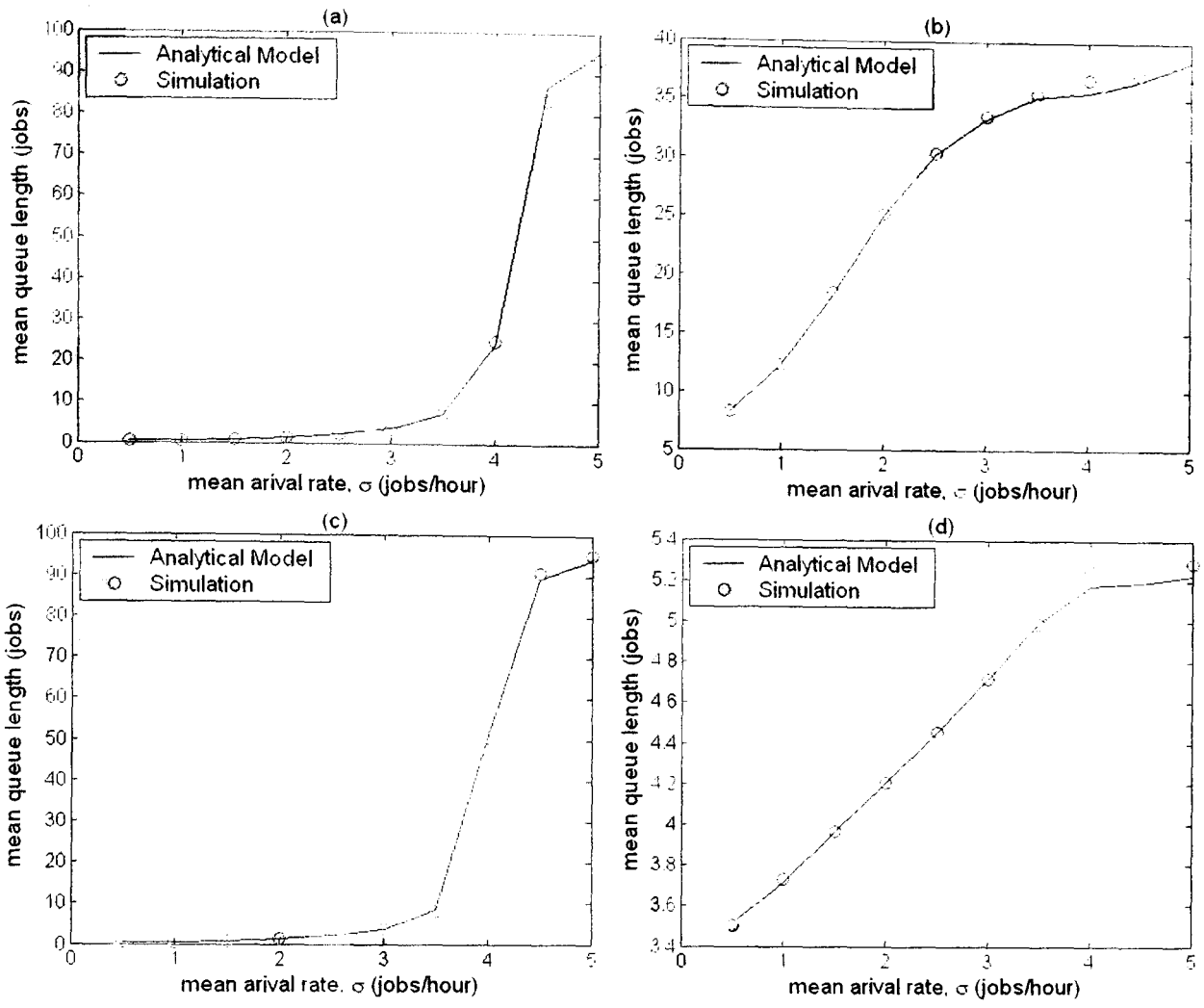

Figure 7.6: MQL_j and MQL_i as a function of σ_1 for tandem networks with multiple servers at the second stage where $\sigma_2 = 3$, $L_i = 40$ and $L_j = 100$((a) $MQL_j, K = 4$, (b) $MQL_i, K = 4$, (c) $MQL_j, K = 8$, (d) $MQL_i, K = 8$).

Computations performed for figures 7.7 and 7.8 show that for performability measure MQL_j the maximum error rate is less then %4.59 and for MQL_i it is less then %3.96 when analytical and simulation results are compared.

Overall results presented in figures 7.4-7.8 show that the new approach is accurate even for relatively large number of servers at the second stage of the tandem network.

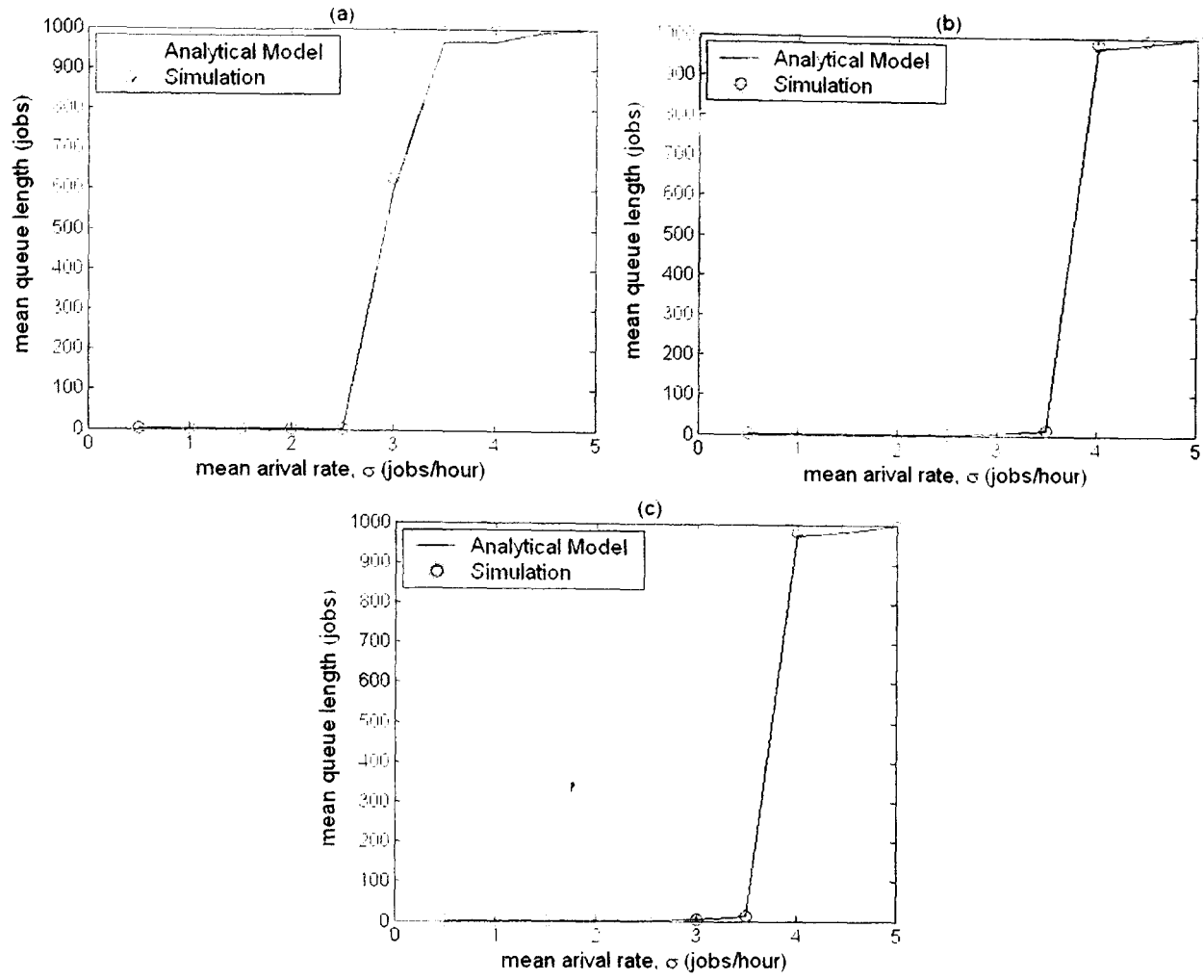

Figure 7.7: MQL_j as a function of σ_1 for tandem networks with multiple servers at the second stage where $L_i = 15$ and $L_j = 1000$((a) $K = 4$, (b) $K = 8$, (c) $K = 12$).

A case study is presented in the following section, where there is a single server at the first stage and multiple servers at the second stage. Such a model is successfully used for modelling Network Memory Servers (NMS). Details of the NMS are given in the following section together with the model used.

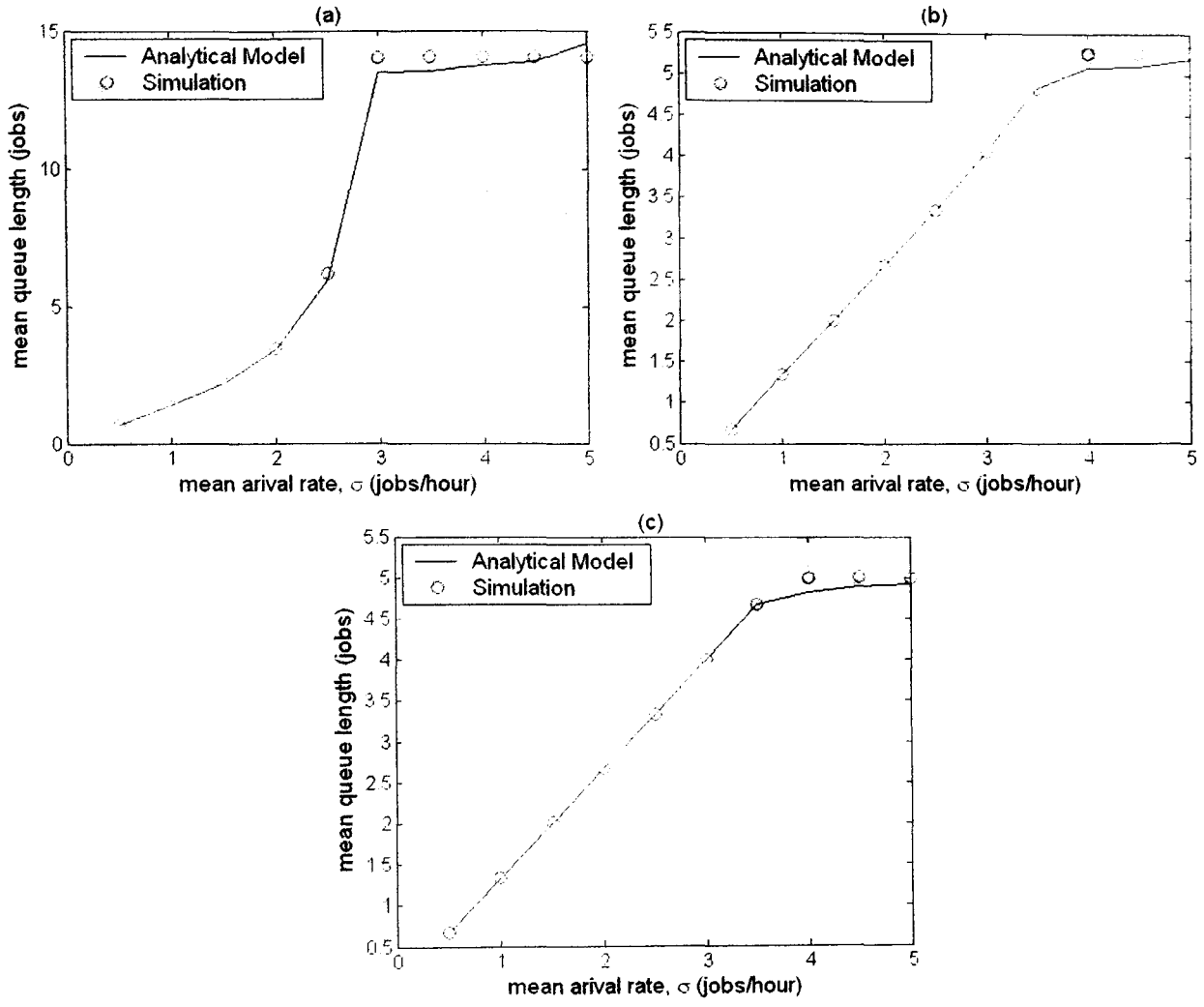

Figure 7.8: MQL_i as a function of σ_1 for tandem networks with multiple servers at the second stage where $L_i = 15$ and $L_j = 1000$ ((a)$K = 4$, (b) $K = 8$, (c) $K = 12$).

7.4 Network Memory Servers with Parallel Processors, Breakdowns and Repairs

Rapid increase in computing power and network speeds enables the development of new types of network services. A Network Memory Server (NMS) is one such service which is being developed at Middlesex University (Mapp et. al. 2004).

The aim of that project is to use Dynamic Random Access Memory (DRAM) and build large memory servers. Since the cost of DRAMs is continuously falling over the last years to build such a system can be affordable. Also since the network speeds are also increasing it can be possible to have faster access to the memory of another machine on a High-Speed Local Area Network or HSLAN than accessing the local hard disk (Mapp et. al. 2004).

Since it is possible to build a new global storage model based on specialised servers, at Middlesex University network memory servers are being developed as a part of an overall architecture to provide central storage facilities for mobile environments (Mapp et. al. 2004).

The performance study of such a system has previously been reported in (Gemikonakli et. al. 2006). However, in this study, multiple servers and breakdowns are not taken into acount. A single NMS with no back-up, and no breakdowns was considered. This was due to the state space explosion problem. In this section, the three dimensional solution approach proposed is used to evaluate the performability of two processor network memory servers with breakdowns, and repairs. The model allows detailed analysis to be performed on the interaction between the server and the network.

The results are generated when the throughput of the network is relatively higher than the throughput of the server which would be the case in Gigabit networks. The cases where the throughput of the network is similar to that of the server are considered as well. This condition is likely to be true in wireless networks which offer lower bandwidth and increased latency compared to wired systems. The model is also attempting to look at the effect of other LAN traffic on the performance of the NMS, since for both wired and wireless cases, the main interest are the systems based on Ethernet technology. Requests from different clients form a distributed network queue which can be modelled as a large capacity queue with exponential service times.

The NMS is transparent to end users of the machines in the system, and since it is stateless it treats requests independently and keeps no record of previous interactions with its clients. Also the NMS deals only with blocks of data. It does not consider the way data is referenced and accessed. The NMS receives requests to create, read, write or delete blocks of data of various sizes. Requests from clients are sent to the NMS using TCP/IP which provides

reliable communication between processes on different machines. The requests to the server along these connections form a distributed input queue. The server uses the socket abstraction to handle each connection. Each socket has a finite buffer associated with it. When this buffer is full, no more requests can be sent to the server which can be modelled as the network no longer serving requests to the NMS (stage one does not forward the messages to stage two if the queue at the second stage is full)(Gemikonakli et. al. 2006).

The requests are kept in the queue of the network until those which are already in the queue receive service. The TCP windowing mechanism prevents packets from being thrown away when the buffer is full and this makes it possible to model the system as a finite queue. Each connection, which represents a client, is managed by a thread and requests from clients are handled by that thread in a sequential fashion. Threads are scheduled before they can service clients' requests. That means there is a distributed contention system at the first stage which can be modelled as a single server with exponential service times. Each client can send a packet containing multiple requests to the NMS. That means for analytic purposes, it is possible to treat such a packet as one job. A global memory storage unit is used which is accessible by all the NMS servers using shared memory techniques (the common queue at the second stage). Reading and writing involves the same basic operation which is taking a request and finding the corresponding block. If the request is a write request then data is written from the incoming network buffer to that block in memory. If a read operation is requested then the data from the block is copied to an outgoing network buffer (Gemikonakli et. al. 2006).

The LAN and the memory server can be thought as two tandem queues each serving arriving jobs. The model given in figure 7.1 can be used to represent such a system. In terms of volume, most of the traffic going to the server will consist of requests to write blocks. This is because the request headers of the NMS are quite small compared to the data being transferred, so write requests will cause a lot of data to be transferred from the client machines to the NMS. In figure 7.1, the rate of this traffic is given as $\mu_1(1-\theta_1)$. On the other hand most of the traffic moving from the NMS to client machines are due to read requests. This is modelled as a feedback loop going from the NMS Server back to its clients and the rate is given as $\mu_2\theta_2$. The LAN is assumed to be highly reliable. The memory servers are prone to failures. When the common queue for the NMSs is full, or both servers are broken, the LAN stops serving this queue, and resumes service when spaces are free in the queue and at least one of the servers is ready to accept more jobs.

The model and the solution presented in the previous section is used to evaluate the performance of the proposed system for specific cases. Breakdown and repair rates have been

taken as $\xi = 0.001$/hr and $\eta = 0.5$/hr respectively. A finite set of values for θ_1 and θ_2 are used which show the share of read and write operations carried out by the NMS respectively.

Figures 7.9 and 7.10 show the MQL and PJL respectively ((a) shows loaded systems, (b) shows systems with relatively lighter loads) for the first stage (i.e. the LAN) as a function of σ_1, where $\sigma_2 = 0$ jobs/hr, $\mu_1 = 10$ jobs/hr, $\mu_2 = 2$ jobs/hr, $\theta_2 = 0.75$, $L_i = 40$ and $L_j = 1000$.

For a loaded network, for large θ_1, the presence of a back-up server becomes less significant. Also the results show that NMS reacts badly to large delays in the system. This is because in most cases clients will be blocked, unable to proceed unless the requested block is retrieved. In this regard, the mean queue lengths are of particular interest. The results suggest that system performance is affected by high values of θ_1, (i.e. the network memory server is competing with other traffic for the network hence it is underutilised). The results also clearly highlight the fact that parallel operation with backup and repair capabilities does improve the general performance of the system. This was observed more prominently for lower values of θ_1 and emphasizes the need to ensure that θ_1 remains relatively small in such environments.

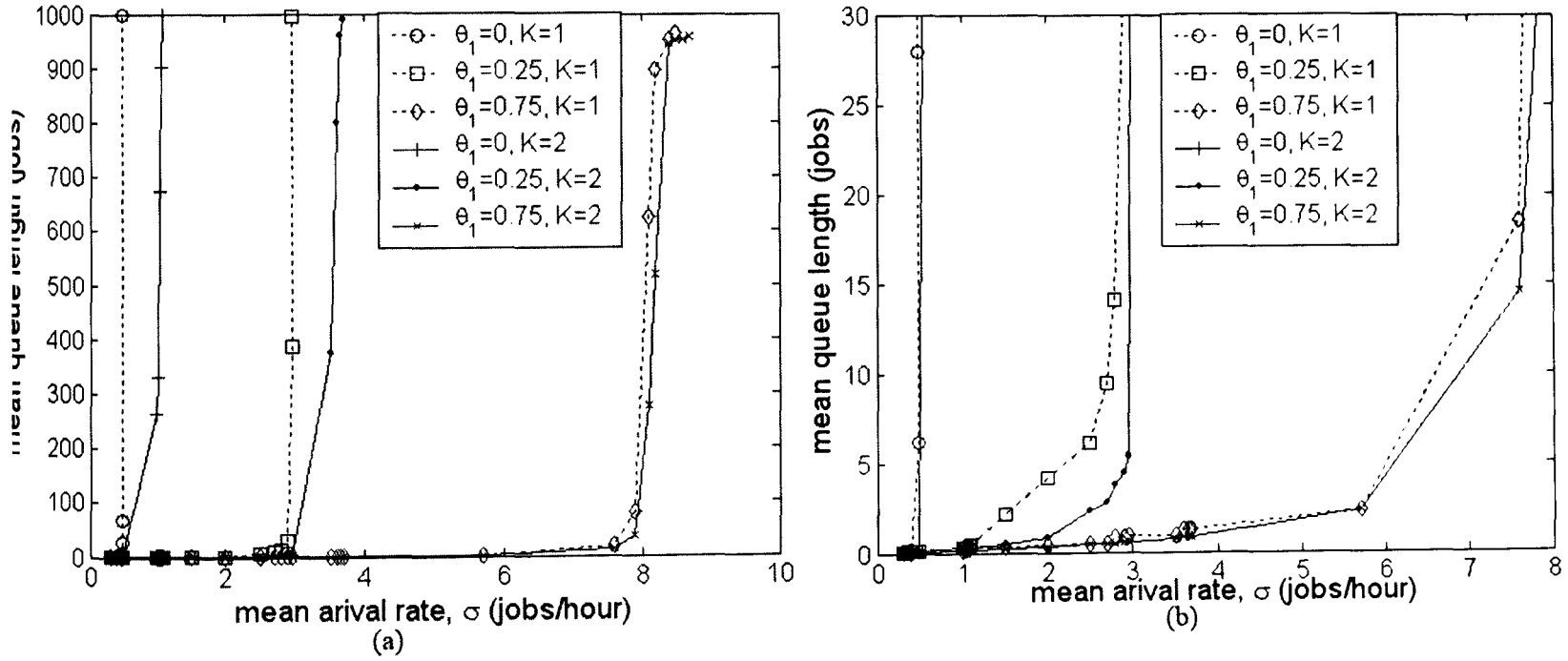

Figure 7.9: MQL for the LAN, for $K = 1$, $K = 2$ and $L_i = 40$

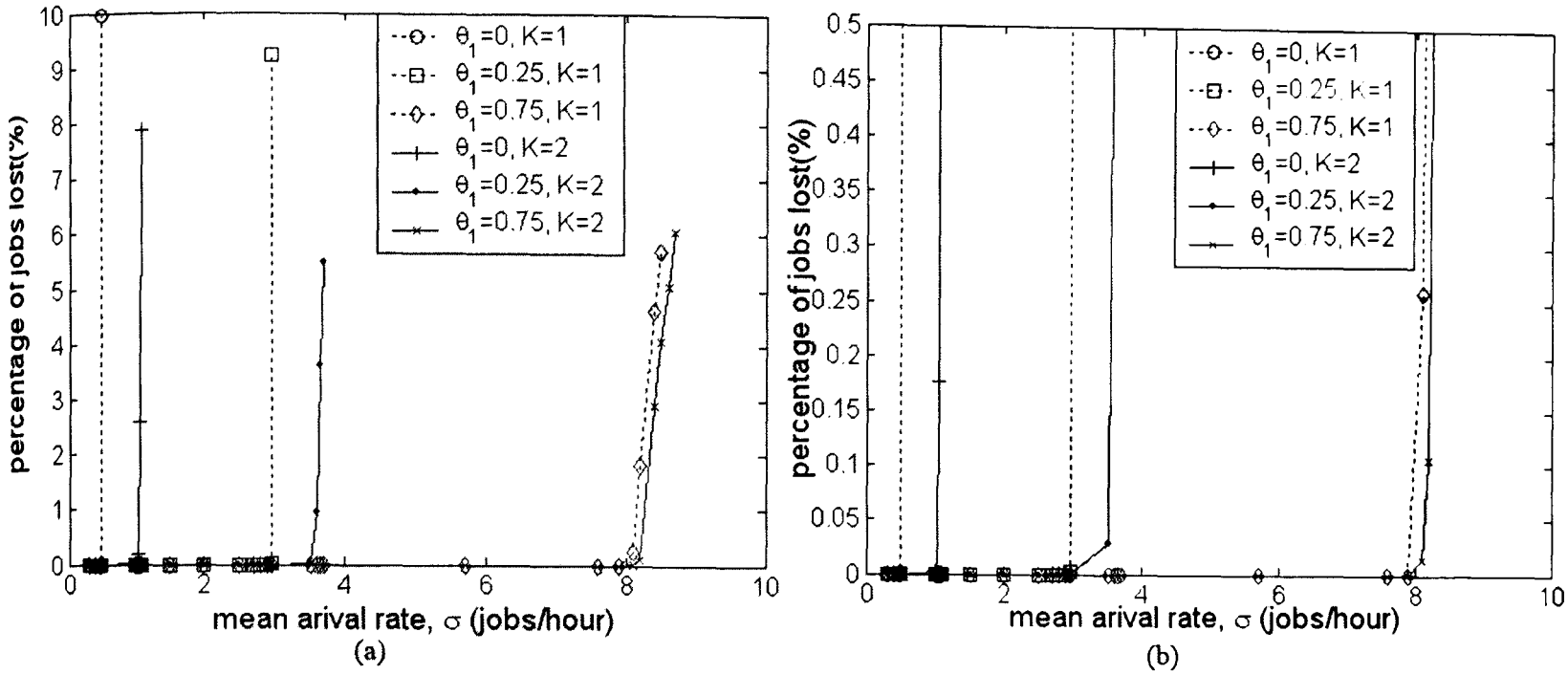

Figure 7.10: PJL for the LAN, for $K = 1$, $K = 2$ and $L_i = 40$

The proposed three dimensional solution is slower than most numerical techniques especially for heavily loaded systems, however the new approach is much faster than simulation and more importantly, unlike most numerical techniques, it can handle larger state space. Furthermore, the use of three dimensions increases the flexibility of the solution making it possible to use known methods (e.g. Spectral Expansion) for each X_k with appropriate transitions complementing the solution. The method is applicable to finite queues only, but very large queuing capacities can be used at the first stage. The CPU time will increase with L_i, L_j and K. A jump in CPU times will occur when the network approaches instability. The technique can further be extended to apply to computing clusters reached through a LAN, where state explosion problem is a serious issue for numerical techniques such as Spectral Expansion.

The approach presented here is applicable to systems presented in chapters 3, 4, and 6 provided that the systems are modelled together with a LAN proceeds the system concerned.

Chapter 8

Conclusion

This thesis describes a work on performability modelling of various multi-server systems. Since multi-server systems with breakdowns and repairs are considered, performability which is the composite measure of performance and availability, is the most realistic modelling approach. Two dimensional state space is used to represent the state of the system considered at a given time and the Spectral Expansion method is extensively used for the solution of the two dimensional state space. Numerical results are presented in order to show the effectiveness of the models developed. Some assumptions are made to make analytical modelling possible and in some cases approximate solutions are used rather than exact ones. In order to validate the accuracy of the developed models, simulations are performed for systems with certain assumptions and approximate solutions. The approaches for the evaluation of the performability of multi-server systems presented in this thesis, lend themselves as powerful tools in order to make essential decisions for system design, optimise parameters to fit the user requirements, and to examine optimal design tradeoffs. Also, these models are helpful for understanding the complex interaction between systems' components and subtle effects of various factors such as failures, repairs and delays.

Models developed and their solutions highlighted the scale of the state space explosion problem inherent to analytical approaches using two dimensional Markov state representations. This problem is widely covered in the literature. Although there are some approaches leading to approximate solutions, these are not generally applicable to two stage tandem networks with feedback and various traffic loads. This problem restricts the size and complexity of network systems that can be considered. In an attempt to ease this problem, a novel approach has been developed and presented in chapter 7. This approach enables the use of parallel processors at the second stage of a tandem network ensuring that an increase in the number of processors does not create state space explosion. Also, this approach makes it possible to consider such systems for performability measures rather than pure perfor-

mance measures since it is possible to handle multiple operative states at the second stage. Case studies showed that performance measures computed have excellent agreement with simulation results. This approach is the main contribution of this thesis.

In addition to this, models presented throughout the thesis emerged as useful models in evaluating the performability of various complex, multi-server systems. The approaches developed can be used to understand the complex architecture of the multi-server systems considered, the subtle effects of various parameters on performability measures, and the interrelationships of these parameters. Furthermore, these analytical approaches are cost effective and hence can be used to identify key parameters and critical components of systems considered. The key contributions of each model are summarised in the next section.

8.1 Contributions of the Thesis

The major contributions of the thesis can be summarised as follows:

1. In Chapter 3 analytical models are presented for homogeneous and a specific type of heterogeneous multi-server systems with reconfiguration and rebooting delays.

   For performability evaluation of homogeneous multi-server systems with reconfiguration and rebooting delays, models considered in (Trivedi et. al. 1990) are extended. The state probabilities in the case of a homogeneous multi-server system with breakdowns, repairs, reconfiguration and rebooting delays are derived using the Spectral Expansion method. Two dimensional representation of the homogeneous multi-server systems with reconfiguration and rebooting delays makes it possible to consider the queuing characteristics of the systems, since one of the random variables is used to represent the number of jobs in the system. Numerical results have been obtained and presented for various performability parameters, for both bounded and unbounded systems. The limiting effects of finite queuing capacities are analysed in detail. Also, since some of the previous studies assume that there are no reconfiguration and/or rebooting delays when a failed server is being mapped out from the system, in this work the effects of various reconfiguation, rebooting and probabilities of having a cover facility is analysed in detail. Results show that, when queue limit is not an important factor on mean queue length performance, the choice of the optimum number of servers depends on the values of these parameters. However, for bounded queuing systems, the queue capacity is the main factor affecting the performability of the system especially when relatively high arrival rates are expected.

In the second part of Chapter 3 a specific type of multi-server system with breakdowns, repairs, and reconfiguration and rebooting delays have been modelled for exact solution. Heterogeneous multi-server systems with one main and several identical servers have been considered and a performability model of these systems is developed. Numerical results have been obtained and presented for various performability parameters, for both bounded and unbounded queuing systems. The results show that for heterogeneous multi-server systems with one main and several identical servers, the discussion should go beyond whether a processor should be added or service rate of the main processor should be improved. Depending on various parameters, such as queuing capacity, reconfiguration and/or rebooting delays, and cover probability, an informed choice can be made. Similar analysis to what is shown in this study will provide the decision makers with sufficient information in deciding whether a processor should be added to the system, service rate of the main processor or cover should be improved, or queue size should be increased.

2. The analytical models used in Chapter 3 are extended in Chapter 4 for modelling farm paradigm multi-server systems with breakdowns, repairs, and reconfiguration and rebooting delays for exact solution. Beowulf clusters are taken as case studies. Typical and highly available Beowulf clusters with one head and several identical processors have been studied. These systems have been considered in (Leangsuksun et. al. 2003, Leangsuksun et. al. 2004) for pure availability measures. Farm paradigm multi-server systems have not been considered for performability modelling. The steady state probabilities for these systems with breakdowns, repairs, reconfiguration and rebooting delays are derived using the QBD processes coupled with the Spectral Expansion method. Simulation results are presented comparatively in order to validate the accuracy of the models presented.

   Numerical results have been obtained and presented for various performability measures, for both bounded and unbounded queuing capacities. The effects of having head nodes with or without backups are analysed as well. Results show that, having a serving head node can have a significant impact on the system performability even if the number of serving processors is kept the same in total. Also results obtained for highly available Beowulf clusters show that having backup processors for the head node can improve the systems' performability significantly. Systems with hot and cold standby processors are also compared. The models developed are highly flexible and they can be used for performability evaluation of typical and highly available Beowulf clusters with various characteristics such as systems with serving or non-serving head

nodes, systems with components which have various reliabilities and various service rates. These numerical results and the methodology are useful to modify/improve the operating system in use and operational aspects effectively, according to the performability requirements. The method is also useful for the designers and users of small scale Beowulf systems which are frequently used by scientific and computing communities. Since Beowulf clusters usually consist of off-the-shelf computers, the performability models are useful in order to specify the system requirements before the cluster is implemented.

3. In Chapter 5 two different approaches, IPP and joint state approaches are extended for performability evaluation of open queuing networks with relatively small number of nodes, in order to consider bounded queues in each node. Solution techniques are presented for performability analysis of open queuing networks with server breakdowns, repairs, external job arrivals and finite queuing capacities. Numerical results have been presented for three node systems. Joint state model, IPP model and simulation results are also presented for validation and comparison purposes. Both joint state and IPP modelling approaches are reasonably accurate for feed forward open networks. However, when open network systems with feedbacks are considered the numerical results obtained from joint state approach provides better approximations. The given approach is applicable for open queues where each node has same queuing capacity as well as nodes with heterogeneous queuing capacities. Such an approach is useful especially for optimisation of queue capacities at each node.

4. Performability studies for multi-server systems using deferred repair strategies exist in the literature. However, queuing issues are not considered in these studies. In Chapter 6, two dimensional representation of homogeneous and HA farm paradigm multi-server systems are considered for performability modelling. Queuing characteristics can be analysed in detail since one of the random variables is used to represent the number of jobs in the system considered.

   For homogeneous multi-server systems, models are presented for systems without reconfiguration and rebooting delays as well as systems with reconfiguration and rebooting delays. The state probabilities of such systems are derived using the Spectral Expansion method. The queuing issues are considered, investigated and analysed in detail. Numerical results have been obtained and presented for various performability parameters, for both bounded and unbounded systems. Effects of having various threshold values to start deferred repairs, deferred repair rates and reconfiguration/ rebooting delays are analysed. Results show that, the performability of a system depends on

the threshold value and deferred repair rate. Also the effects of deferred repair rate increase as the failure rate of the systems increases. The analytical models developed can be used in order to specify the optimum threshold values where the deferrals of the repairs are affordable in terms of performability.

In the remaining sections of Chapter 6, highly available clusters with breakdowns, deferred repairs, and switching delays have been modelled for exact solution. These systems have not been considered for performability modelling in the literature. The queuing issues are considered, investigated and analysed in detail for these systems as well. Effects of having various threshold values to start deferred repair, various deferred repair rates, and various switching delays are analysed. Both hot and cold standby techniques are considered while switching delays are chosen.

The given models can be used to compare the cost of deferred repairs in terms of performability degradation to the cost of traditional repair strategies for systems with various characteristics. The models presented are flexible and useful for computer and communication systems with deferred repairs and queuing considerations.

5. In Chapter 7 a new approach is presented in order to ease the state explosion problem in two stage tandem networks. The new approach uses two dimensional state space to represent the number of jobs in each state. Each operative state of the servers at the second stage is presented in a finite lattice strip and Spectral Expansion method is used in order to solve these lattice strips for state probabilities. Once the state probabilities in each lattice strip is computed an iterative solution is employed in order to include the transitions between operative states of the servers which are presented at the third dimension.

Although the proposed three dimensional solution is slower than most numerical techniques especially for heavily loaded systems, this new approach is much faster than simulation and more importantly, unlike most numerical techniques, it can handle larger state space. Comparative results show that unlike other solutions such as Spectral Expansion, the state space limitations become independent of the number of servers at the second stage. Also comparisions with simulations show that the new approach is accurate even for large number of servers at the second stage of tandem networks.

Another advantage of the new method is the flexibility it provides in terms of transition matrices. Since the Spectral Expansion method is used for each operative state of the servers, transition matrices can be modified according to the scenario at each operative

state. For instance, it is possible to stop stage one to send jobs to stage two if there are no operative servers at the second stage. The existing solution methods are not able to handle such scenarios. A case study is considered for NMS which is a multi-server system implemented in Middlesex University and such a scenario is implemented for the cases where all of the servers of NMS are broken. In (Gemikonakli et. al. 2006) a similar study is performed for NMS systems but server failures and multiple servers could not be considered because of the numerical difficulties caused by large number of states.

8.2 Suggestions for Future Study

Some suggestions for future study are given below

1. State space models with multiple components suffer from state explosion problem since the functional equations arising in the analysis of such processes usually cause analytic difficulties (Boxma et. al. 1994). This problem is common for models representing 2-dimensional Markov processes as well. It is possible to find approximate solution methods which give accurate results to ease such problems. An approximate three dimensional solution is provided in Chapter 7 for two stage tandem networks. It is possible to develop a similar approach for multi-server systems as well. Such an approach would be very useful especially for large scale clusters. The models presented in this thesis can be solved with such an approximate approach in future studies. The studies given for homogeneous multi-server systems in (Gemikonakli et.al 2007) look promising enough to provide approximate solution for homogeneous multi-server systems with large numbers of servers.

2. The method given in Chapters 3 and 4 can be further extended to many of the high performance, highly available, highly reliable computer architectures, with appropriate modifications. The same approach can be used in modelling various kinds of cluster systems.

   The two dimensional representation of models with reconfiguration and rebooting delays can be extended to model flexible manufacturing cells for performability measures, hence the model becomes highly relevant to manufacturing or production research.

3. The method presented for multi-server systems with deferred repairs can be extended for systems with various characteristics. Since in the given approaches the states where the repairman is present and states where the repairman is at a remote location are

presented separately, systems with complex architecture may require large number of states. In this case an approximate solution method can be employed in order to ease the numerical difficulties caused by large number of states.

4. The approximate solution given for two stage tandem networks can be applied for various scenarios. For NMS the transition matrices are modified so that the local area network does not send any jobs to NMS if none of the servers at NMS are operative. The model provides a large degree of flexibility and various scenarios can be handled as long as the system is stable and the Spectral Expansion method is able to handle the two dimensional lattice strips.

Bibliography

[1] Adams, J. and D. Vos (2002), "Small-College Supercomputing: Building A Beowulf Cluster At A Comprehensive College'l". In Proceedings of 33rd SIGCSE Technical Symposium on Computer Science Education. Cincinnati, Kentucky. pp:411-415.

[2] Amari, S. V., G. Dill and E. Howald. (2003), "A New Approach to Solve Dynamic Fault Trees", In Proceedings of Annual Reliability and Maintainability Symposium, pp:374-379.

[3] Bader, D.A., B.M.E. Moret and P. Sanders (2002), "High-Performance Algorithm Engineering for Parallel Computation," Experimental Algorithmics, Lecture Notes of Computer Science, 2547, pp.1-23.

[4] Banks, J., J. S. Carson, B. L. Nelson and D.M. Nicol. (2005), Discrete-Event System Simulation, (4th. ed.) Prentice-Hall: Upper Saddle River, NJ.

[5] Baskett, F., K.M. Chandy, R.R. Muntz and F. Palacios-Gomez (1975), "Open, Closed and Mixed Networks of Queues with Different Classes of Customers", Journal of the ACM, vol. 22, pp: 248-260.

[6] Beaudry, M. D. (1978), "Performance-related Reliability Measures for Computing Systems". IEEE Trans. Computer, vol. C-21, pp: 540-547 ,June.

[7] Bossen, D. C., A. Kitamorn, K. F. Reick and M. S. Floyd. (2002), "Fault-Tolerant Design of the IBM pSeries 690 System UsingPOWER4 Processor Technology," IBM J. Res. & Dev. vol: 46(1), January.

[8] Boxma, O.J., G.M. Koole and Z. Liu. (1994), "Queueing-theoretic Solution Methods for Models of Parallel Distributed Systems". Performance Evaluation of Parallel and Distributed Systems, eds. (CWI Tract 105, Amsterdam,), pp. 1-24.

[9] Brim, M.J., T. G. Mattson , S. L. Scott, (2001), "OSCAR: Open Source Cluster Application Resources" In Ottawa Linux Symposium 2001, Ottawa, Canada

[10] Brown, R. G. (2004), "Engineering a Beowulf-Style Compute Cluster", Online Gnu Open Publication License Book provided to the Beowulf Community *Available at : http : //www.phy.duke.edu/brahma/Resources/beowulf_book.php*2004

[11] Brown, R. G. (2007), "Introduction to the Beowulf Design", Duke University Physics Department Durham, NC 27708-0305 Copyright Robert G. Brown,

[12] Buzacott, J. A. and J.G. Shanthikumar (1993), Stochastic Models of Manufacturing Systems, Prentice-Hall.

[13] Carrasco, J. A. (2006), "Adapted Importance Sampling Schemes for the Simulation of Dependability Models of Fault-Tolerant Systems with Deferred Repair". Proceedings of 39th Annual Simulation Symposium, Huntsville, AL, USA.

[14] Chakka, R. and I. Mitrani (1992), "A Numerical Solution Method for Multiprocessor Systems with General Breakdowns and Repairs". In Proceedings of the 6th International Conference Modelling Techniques and Tools, Edinburgh, UK, pp: 289-304.

[15] Chakka, R. and I. Mitrani, (1994), "Heterogeneous Multiprocessor Systems with Breakdowns: Performance and Optimal Repair Strategies", Theoretical Computer Science, vol.125, pp: 91-109.

[16] Chakka, R. (1995), Performance and Reliability Modelling of Computing Systems Using Spectral Expansion, PhD Thesis, University of Newcastle Upon Tyne, UK.

[17] Chakka, R. and I. Mitrani.(1996), "Approximate Solutions for Open Networks with Breakdowns and Repairs", in the book "Stochastic Networks: Theory & Applications", eds: F. P. Kelly, S. Zachary and I. Ziedins , Oxford University Press, September, Royal Statistical Society Lecture Note Series No. 4.

[18] Chakka, R. (1998), "Spectral Expansion Solution for Some Finite Capacity Queues", Annals of Operations Research, vol. 79, pp: 27-44.

[19] Chakka, R., O. Gemikonakli and P.G. Harrison (2000), "Approaches to Modelling Open Networks with Bursty Arrivals", Eighth IFIP Workshop on Performance Modelling and Evaluation of ATM & IP Networks, Ilkley, vol. 13. pp: 1-11.

[20] Chakka, R., O. Gemikonakli, and P. Basappa (2002), "Modelling Multiprocessor Systems with Time or Operation Dependent Breakdowns, Alternate Repair Strategies, Reconfiguration and Rebooting Delays". In Proceedings of SPECTS 2002, pp: 266-277.

[21] Ciardo, G. and E. Smirni (1999), "ETAQA: An Efficient Technique for the Analysis of QBD-processes by Aggregation," Performance Evaluation, vol. 36-37, pp: 71-93.

[22] Ciardo G., R. A. Marie, B. Sericola, and K. S. Trivedi (1990), "Performability Analysis Using Semi-Markov Reward Processes". IEEE Transactions on Computers, vol. 39(10), pp: 1251-1264.

[23] Cohen, J.W. (1995), "On a Class of Two-Dimensional Nearest-Neighbour Random Walks." In: Studies in Applied Probability - Papers in honour of Lajos Takács, eds. J. Galambos and J. Gani, J. Appl. Probab. vol. 31A, pp: 207-237.

[24] Cotroneo D., G. Paolillo, S. Russo and M. Lauria (2005), "CSAR-2: A Case Study of Parallel File System Dependability", 2005 International Conference on High Performance Computing and Communications (HPCC-05), Sorrento, Italy, pp. 180-189, 21-24 September.

[25] Dayar, T.(1998), "State Space Orderings for Gauss-Seidel in Markov Chains Revisited", SIAM Journal on Scientific Computing vol.19 , pp:148-154.

[26] Deitel, M. Harvey (1990), "An Introduction to Operating Systems". Boston: Addison-Wesley.

[27] Dudin, A.N., A.A. Shaban, and V.I. Klimenok (2005), "Analysis of a BMAP|G|1|N queue". International Journal of Simulation: Systems, Science and Technology. vol. 6, No1-2, pp: 13-23.

[28] El-Rayes, A., M. Kwiatkowska and G. Norman (1999), "Solving Infinite Stochastic Process Algebra Models Through Matrix-Geometric Methods". Proc. 7th Process Algebras and Performance Modelling Workshop (PAPM'99), edrs: J. Hillston and M. Silva, University of Zaragoza, pp: 41-62, September.

[29] Engelmann, C. and S. L. Scott. (2005), "Concepts for High Availability in Scientific High-End Computing". Proceedings of High Availability and Performance Computing Workshop (HAPCW), Santa Fe, NM, USA, October.

[30] Fang, Y-C., S. Iqbal and A. Saify (2005), "Designing High-Performance Computing Clusters", Technical Report. *Available at : www.dell.com/powersolutions*, Reprinted: Dell Power Solutions, May. Copyright © 2005 Dell Inc. All rights reserved.

[31] Fiems, D., B. Steyaert and H. Bruneel (2004). "Discrete-time Queues with Generally Distributed Service Times and Renewal Type Server Interruptions", Performance Evaluation, vol. 55, pp:277-298.

[32] Fischer, W. and K. Meier-Hellstern (1993), "The Markov-Modulated Poisson Process", Performance Evaluation, vol. 18(2), pp:149-171, September.

[33] Fransisco, A. and M. Akhtar (2003), "Grid Computing Environment using a Beowulf Cluster". American Physical Society, Texas Section Fall, October 23-25, Texas Tech Campus, Lubbock Texas, MEETING ID: TSF03, Abstract #B1.005.

[34] García, C., Rubén S. Montero, Manuel Prieto, Ignacio Martín Llorente and Francisco Tirado (2002), "Beowulf Performance in CFD Multigrid Applications". 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing (EUROMICRO-PDP 2002), p. 0007 IEEE.

[35] Gelenbe, E.(1991). "Product-Form Queueing Networks with Negative and Positive Customers". Journal of Applied Probability, vol.28. pp: 656.

[36] Gemikonakli, O., G. Mapp, D. Thakker and E. Ever (2006), "Modelling and Performability Analysis of Network Memory Servers", 39th Annual Simulation Symposium, Huntsville, Alabama, USA, pp.127-134, 2nd - 6th April.

[37] Gemikonakli, O., H. Sanei, and E. Ever (2007), "Approximate Solution for the Performability of Markovian Queuing Networks with a Large Number of Servers", Proceedings of the 5th International Workshop on Signal Processing for Wireless Communication (SPWC 2007), King's College London, 6-8 June.

[38] Goyal, A., S. S. Lavenberg and K. S. Trivedi (1987), "Probabilistic Modeling of Computer System Availability", Ann. of Oper. vol. 8, pp: 285-306.

[39] Grove, D.A. and P.D. Coddington (2005),"Communication Benchmarking and Performance Modelling of MPI Programs on Cluster Computers", Journal of Supercomputing, vol. 34, pp.201-217.

[40] Guest, M.F. (2001), "Application Performance on Beowulf Cluster Systems", Technical Report of UKHEC Collaboration, *Available at : http : //www.ukhec.ac.uk/research/beowulf.UKHEC.pdf*.

[41] Gyu, S. C., J. H. Kim, D. Ersoz, A. Yoo and C. R. Das (2004), "Coscheduling in Clusters: Is It a Viable Alternative?", Proceeding of Super Computing (SC), Pittsburgh, Pennsylvania, USA.

[42] Hacker, T. J. and B. D. Athey (2001), "A Methodology for Account Management in Grid Computing Environments", In proceedings of GRID 2001, Denver, Colorado, USA, pp: 133-144.

[43] Haddad, I., C. Leangsuksun and S.L. Scott (2003), "HA-OSCAR: the birth of highly available OSCAR". Linux Journal, vol.1, pp: 115.

[44] Harrison, P. G. and N.M. Patel (1993), Performance Modelling of Communication Networks and Computer Architecture: Addison-Wesley.

[45] Harrison, J. M. and R. J. Williams (1987), "Brownian Models of Open Queueing Networks with Homogeneous Customer Populations". Stochastics, pp: 77-115.

[46] Haverkort, B. R. and A. Ost (1997), "Steady-State Analysis of Infinite Stochastic Petri Nets: Comparing the Spectral Expansion and the Matrix-Geometric Method", Proceedings of the Seventh International Workshop on Petri Nets and Performance Models, Saint Malo, France, pp: 36-45.

[47] Hawick, K.A., D. A. Grove, P. D. Coddington and M. A. Buntine (2000), "Commodity Cluster Computing for Computational Chemistry". Internet Journal of Chemistry, ISSN: 1099-8292, Article 4.

[48] Heidelberger, P .and K.S. Trivedi (1982), "Queuing Network Models for Parallel Processing with Asynchronous Tasks". IEEE Transactions on Computers, vol.31, pp:1099.

[49] Henty, D. (2000), "The Grid A Critical Review of Current Status and Future Directions in Grid Technology the DIRECT Initiative", EPCC.

[50] Horton, G.and S.T. Leutenegger (1994), "A Multi-Level Solution Algorithm for Steady-State Markov Chains", SIGMETRICS, pp: 191-200.

[51] Jain, R. (1991), The Art of Computer Systems Performance Analysis, John Wiley & Sons, Inc.

[52] Kelly, F.P. (1996), "Modelling communication networks, present and future", Proceedings of Philosophical Transactions of the Royal Society, London, U.K., pp: 437-463.

[53] Klimenok, V.I., A. V. Kazimirsky, A. N. Dudin, L. Breuer and U. R. Krieger (2005), "Queueing Model MAP|PH|1|N with Feedback Operating in the Markovian Random Environment", Austrian Journal of Statistics, vol.34(2), pp:101-110.

[54] Law, A. and W.D. Kelton (2000), Simulation Modelling and Analysis, 3rd ed., McGraw-Hill: NY, USA.

[55] Leangsuksun, C., L. Shen, T. Liu, H. Song and S. Scott (2003), "Dependability Prediction of High Availability OSCAR Cluster Server", The 2003 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'03), Las Vegas, NV, USA, June 23-26.

[56] Leangsuksun, C., L Chen, T. Lui and S. Scott (2003b), "The Modelling and Dependability Analysis of High Availability OSCAR Cluster System", Proceeding of The 17th Annual International Symposium on High Performance Computing Systems and Applications, Sherbrooke, Canada.

[57] Leangsuksun, C., H. Song, L. Shen. (2003c), "Reliability Modelling Using UML". Proceeding of The 2003 International Conference on Software Engineering Research and Practice, Las Vegas, Nevada, USA, pp: 259-262, June 23-26.

[58] Leangsuksun, C., T. Liu, Y. Liu, S. L. Scott, R. Libby and R. Haddad (2004), "Highly Reliable Linux HPC Clusters: Self-Awareness Approach", In Lecture Notes in Computer Science, Vol. 3358: Parallel and Distributed Processing and Applications: Second International Symposium, (Hong Kong) pp:217-222.

[59] Leangsuksun, C., T. Liu, T. Rao, S. Scott and R. Libby (2004b), "A Failure Predictive and Policy-Based High Availability Strategy for Linux High Performance Computing Cluster", Proceedings of The 5th International Conference on Linux Clusters: The HPC Revolution 2004, Austin, Texas, USA.

[60] Leangsuksun, C., V. K. Munganuru, T. Liu, S.L Scott and C. Engelmann (2005), "Asymmetric Active-Active High Availability for High-end Computing", In proceedings of 2nd international Workshop on Operating Systems, Programming Environments and Management Tools for High-Performance Computing on Clusters, Cambridge. MA. USA.

[61] Li, Q.L., Y. Ying and Y. Zhao (2006), "The BMAP/G/1 Retrial Queue with Server subject to Breakdowns and Repairs", Annals of Operations Research, vol. 141, pp: 253-292.

[62] Mapp, G.E, D. Silcott and D. Thakker (2004), "Network Memory Servers: An idea whose time has come", Multi- Service Networks (MSN), Cosener's House, Abingdon, July.

[63] Meyer, J.F. (1980), "On Evaluating the Performability of Degradable Computing Systems", IEEE Transactions Computer, vol. C-29, pp: 720-131.

[64] Mitrani, I. and R. Chakka (1995), "Spectral Expansion Solution for a Class of Markov Models: Application and Comparison with the Matrix-Geometric Method." Perform. Eval. 23(3), pp: 241-260.

[65] Mitrani, I. (2005), "Approximate solutions for heavily loaded Markov-modulated queues", Perform. Eval. Vol.62 (1-4), pp: 117-131.

[66] Mohamed, A. M., L. Lipsky and R. A. Ammar (2003), "Performance Modelling of a Cluster of Workstations", The 4th International Conference on Communications in Computing (CIC-2003), Las Vegas, NV, pp: 227-233, June 23-26.

[67] Nagaraja, K., G. Gama, R. Bianchini, R. P. Martin, W. Meira Jr., and T. D. Nguyen (2005), "Quantifying the Performability of Cluster-Based Services", IEEE Transactions on Parallel and Distributed Systems, vol. 16(5), May.

[68] Neuts, M.F. (1981), Matrix Geometric Solutions, Johns Hopkins University Press.

[69] Numerical Algorithms Group (2005), "NAG C Library Manual" , NAGNP3660/8December 2005 ISBN:978-1-85206-206-4 (ISBN 1-85206-206-1).

[70] Ny, L. M. and B. Sericola, (2002), "Transient Analysis of the BMAP/PH/1 Queue", International Journal of Simulation: Systems, Science & Technology. Special Issue on Analytical & Stochastic Modelling Techniques, vol.3 (3-4), pp: 4 -14.

[71] Pratt, T.W. (1998), "System Performance Evaluation Project", Center of Excellence in Space Data and Information Sciences (CESDIS), NASA Goddard Space Flight Center.

[72] Reiman, M. I. and B. Simon (1989), "Open queueing networks in light traffic", Mathematics of Operations Research, pp: 26-59.

[73] Righter, R. (1996), "Optimal Policies for Scheduling Repairs and Allocating Heterogeneous Processors", Journal of Applied Probability, vol.33, pp:536-547.

[74] Sahner, R. A. and K. S. Trivedi (1987), "Reliability modeling using SHARPE,", IEEE Trans. Reliability, vol. R-36, pp: 186-193.

[75] Schroeder, B. and G. Gibson (2006), "A Large Scale Study of Failures in High-Performance-Computing Systems", International Symposium on Dependable Systems and Networks (DSN 2006), pp. 249-258.

[76] Seelen, L. P. (1986). "An Algorithm for Ph/Ph/c Queues", European Journal of Operations Research, vol.23, pp: 118-127.

[77] Sheldon, F. T., K. Jerath and S. Greiner (2002), "Examining Coincident Failures and Usage Profiles in Reliability Analysis of an Embedded Vehicle Sub-System", 16th European Simulation Multiconference (ESM'02), Darmstadt, Germany, pp. 558-563, June 3-5.

[78] Smith, R. M., K. S. Trivedi and A. Ramesh.(1988), "Performability Analysis: Measures, An Algorithm and a Case Study". IEEE Transactions on Computers, vol. C-37(4), pp: 406-417.

[79] Song, H., C. Leangsuksun, R. Nassar, N. R. Gottumukkala, S. L. Scott and A. Yoo (2006), "Availability Modelling and Analysis on High Performance Cluster Computing Systems", Proceedings of The First International Conference on Availability, Reliability and Security (ARES 2006), Vienna, Austria, pp: 305-313, April 20-22.

[80] Stecke, K. E. and I. Kim (1989), "Performance Evaluation for systems of Pooled Machines of Unequal Sizes: Unbalancing vs. Blocking", European Journal of Operations Research, vol.42, pp: 22-38.

[81] Stecke, K.E. (1992), "Machine Interference: Assignment of Machines to Operators", Handbook of Industrial Engineering, 2nd Ed, Chapter 57, eds: Gavriel Salvendy

[82] Sterling, T. (2001), Beowulf Cluster Computing with Linux, Publisher: MIT Press Publish Date: October, ISBN: 0262692740, pp: 496.

[83] Sun, H., D. Tang and R. Wood (2005), "Optimizing service strategy for systems with deferred repair", Proceedings of 11th Pacific Rim International Symposium on Dependable Computing (PRDC'05), Changsha, China.

[84] Tang, D. and K. S. Trivedi (2004), "Hierarchical Computation of Interval Availability and Related Metrics", Proceedings of 2004 International Conference on Dependable Systems and Networks (DSN'04), pp: 693.

[85] Temsamani, J. and J. A. Carrasco (2002), "Transient Analysis of Markov Models of Fault- Tolerant Systems with Deferred Repair using Split Regenerative Randomization", Technical Report DMSD 2002, Universitat Politècnica de Catalunya.

[86] Tran, H.T. and T.V. Do (1999), "Comparison of Computational Methods for QBD Processes", Proceedings of the 4th International Conference on Applied Informatics. Eger-Noszvaj, Hungary.

[87] Trivedi, K. S. (2002), Probability and Statistics with Reliability, Queuing, and Computer Science Applications: Wiley, NY, USA.

[88] Trivedi, K. S., A.S. Sathaye, O.C. Ibe and R.C. Howe (1990), "Should I Add a Processor?", In Proceedings of 23rd Annual Hawaii International Conference on System Sciences, IEEE Computer Society Press, pp: 214-221.

[89] Trivedi, K. S., O. C.Ibe, R. C. Howe and A.Aggarwal (1996), "Availability and Performance-Based Sizing of Multiprocessor Systems.", Communications in Reliability, Maintainability and Serviceability: An International Journal, vol. 2(2), pp: 14-23.

[90] Trivedi, K. and M. Xiaomin (2002), "Probabilistic Analysis of Wireless Cellular Networks", Proceedings of International Symposium on Performance Evaluation of Computer and Telecommunication Systems, vol.79, pp: 27-44.

[91] Trivedi, K. and N. Lopez-Benitez (1993), "Multiprocessor Performability Analysis", IEEE Transactions on Reliability, vol. 42 (4), pp:579-587.

[92] Trivedi, K. S, J. K. Muppala, S.P. Woolet and B.R. Haverkort (1992), "Composite Performance and Dependability Analysis", Performance Evaluation, vol. 14(3-4), pp: 197-215.

[93] Wagner, A. S., H.V. Sreekantaswamy and S.T. Chanson (1997), "Performance Models for the Processor Farm Paradigm", IEEE Transactions on Parallel and Distributed Systems, pp: 475 - 489.

[94] Watkins, D.S (2000), "Performance of the QZ Algorithm in the Presence of infinite Eigenvalues", SIAM J. Matrix Anal. Appl., vol.22, pp: 364-375.

[95] Yoo, A. B. and M. A. Jette (2001), "The Characteristics of Workload on ASCI Blue-Pacific at Lawrence Livermore National Laboratory", Proceedings of CCGrid2001, Brisbane Australia, pp: 295-3021.